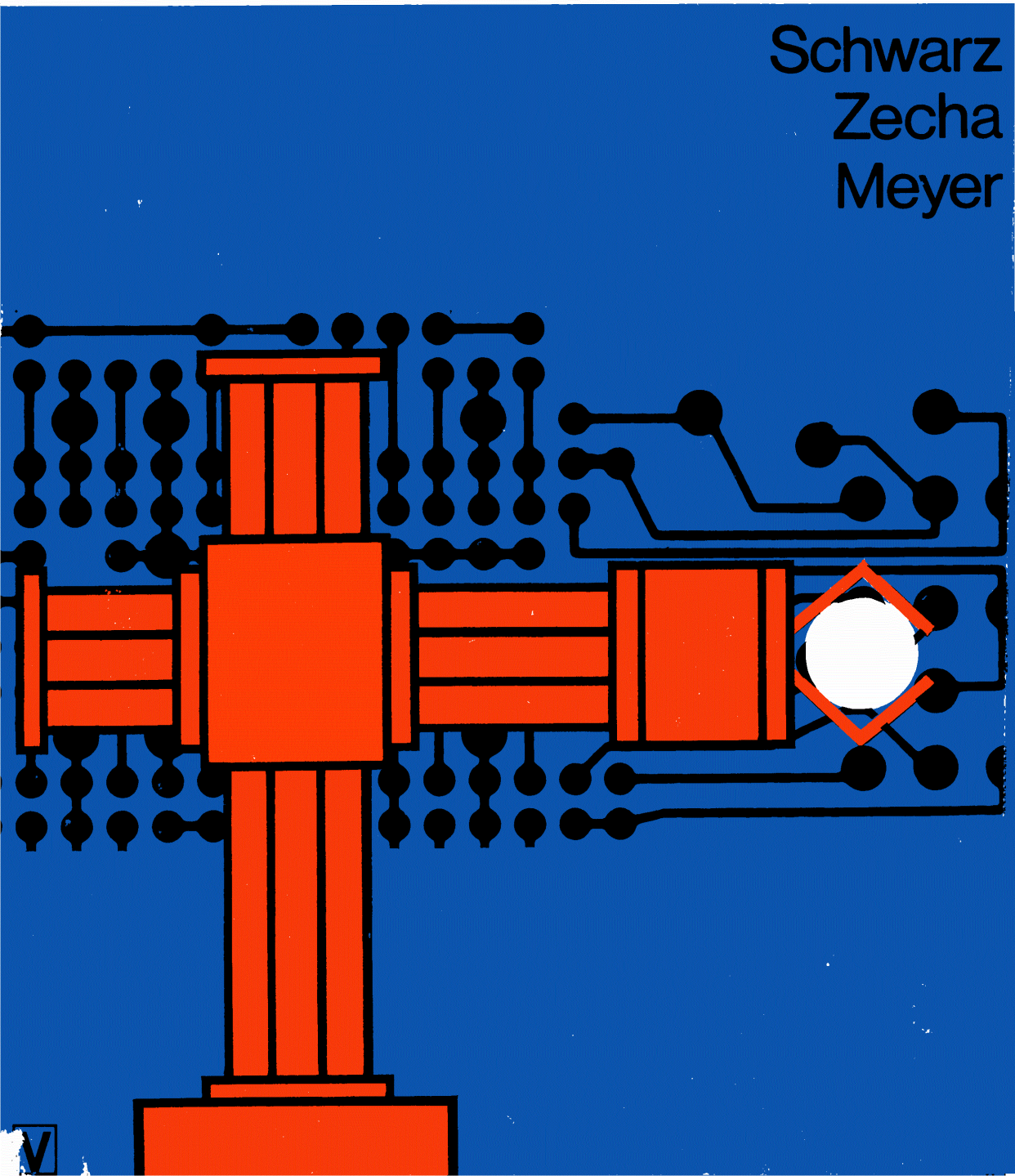
Industrieroboter-steuerungen

Schwarz
Zecha
Meyer



Umfassende Kenntnisse der theoretischen Grundlagen und der zur Verfügung stehenden Gerätetechnik moderner Steuerungssysteme sind das unbedingte Rüstzeug für jeden, der sich mit dem Entwurf und Einsatz moderner Steuerungen für Industrieroboter beschäftigen will. Der Vermittlung von Kenntnissen auf dem für so viele Industriezweige wichtigen Gebiet dient dieses Buch.

Nach einer kurzen Einführung werden die Grundprinzipien der Steuerung von Industrierobotern dargestellt, dabei wird eine Strukturierung des Steuerungssystems vorgenommen und eine Übersicht über die Funktion der einzelnen Teilgebiete gegeben. Besonderer Wert wird dann auf die Behandlung der Modelle der kinematischen Systeme, der Antriebe und Meßsysteme gelegt. Danach wird die Bewegungssteuerung mit ihren beiden Komponenten der Bahnsteuerung und der Servoregelung besprochen. Es folgt eine systematische Darstellung des Entwurfs binärer Ablaufsteuerungen. Bei der Behandlung der Programmierung von Industrierobotern wird der textuellen Programmierung und den höheren Programmiersprachen besondere Aufmerksamkeit gewidmet. Der Sensortechnik und den visuellen Erkennungssystemen für Industrieroboter der 2. und 3. Generation ist ein gesonderter Abschnitt gewidmet. Zum Aschluß werden die beiden in der DDR verfügbaren Industrierobotersteuerungssysteme vorgestellt. Die Ausführungsbeispiele sind für die Praxis von besonderem Interesse.

Das Werk ist für alle diejenigen von Nutzen, die auf dem Gebiet des Entwurfs und des Einsatzes von Industrierobotersteuerungen tätig sind.

Industrierobotersteuerungen

Ingenieurschule
für
Schwerin-Ingenbau
„Walter Ulbricht“
Roßwein
Bibliothek
unbeitig

Autoren

Dr.-Ing. J. Bartsch
(Abschnitt 8.)

Dr.-Ing. S. Fensch
(Abschnitt 5.)

Dr.-Ing. P. Haferkorn
(Abschnitt 8.)

Dipl.-Ing. R. Hiersemann
(Abschnitt 5.)

Dr. rer. nat. J. Hübener
(Abschnitt 6.)

Dipl.-Ing. L. Könitzer
(Abschnitte 8.; A 2.)

Dr.-Ing. K. Littmann
(Abschnitt 4.)

Doz. Dr. sc. techn. W. Manthey
(Abschnitt 3.)

Prof. Dr. sc. techn. G. Meyer
(Abschnitt 5.)

Doz. Dr.-Ing. R. Nollau
(Abschnitte 3.; 4.)

Prof. Dr. sc. techn. W. Schwarz
(Abschnitte 1.; 2.; 3.; 4.; 6.)

Dr.-Ing. B. Weber
(Abschnitt 5.)

Dr.-Ing. H. Wilke
(Abschnitte 8.; A 2.)

Prof. Dr.-Ing. G. Wollenberg
(Abschnitte 8.; A 2.)

Dr.-Ing. M. Zecha
(Abschnitte 3.; 4.; 6.; 7.)

Industrieroboter-steuerungen

Herausgegeben von
Prof. Dr. sc. techn. Wolfgang Schwarz
Dr. sc. techn. Michael Zecha und
Prof. Dr. sc. techn. Gernot Meyer

Ingenieurschule
Schwermaschinenbau
„Walter Ulbricht"
Calbwein
Bibliothek
8294
86/6193
K

VEB VERLAG TECHNIK BERLIN

1. Auflage
© VEB Verlag Technik, Berlin, 1985
Lizenz 201 · 370/62/85
DK 62-50 · LSV 3043 · VT 3/5723-1
Lektor: Jürgen Reichenbach
Schutzumschlag: Kurt Beckert
Printed in the German Democratic Republic
Gesamtherstellung: Offizin Andersen Nexö, Graphischer Großbetrieb, Leipzig III/18/38
Bestellnummer: 553 351 3
03500

Hochschule Mittweida (FH)
University of Applied Sciences
Hochschulbibliothek
Technikumstraße 11
09648 Mittweida

Vorwort

Industrieroboter gelten heute als Symbol fortgeschrittener Technologie. Produktion und Einsatz dieser Automatisierungsmittel haben im letzten Jahrzehnt eine stürmische Entwicklung genommen, wobei ihre potentiellen Einsatzmöglichkeiten bei weitem nicht erschöpft sind. Die Grundeigenschaften, die den Einsatz von Industrierobotern in den verschiedensten Industriezweigen ermöglichen, sind ihre universelle Beweglichkeit und die freie Programmierbarkeit. Während die Bewegungsmöglichkeiten durch den Maschinenbauer zu gewährleisten sind, ist die Realisierung der Programmierbarkeit und die programmgesteuerte Ausführung von Bewegungen eine steuerungstechnische Aufgabe. Industrieroboter sind vorwiegend elektronisch gesteuerte Maschinen.

Dieses Buch soll eine Einführung und Übersicht über die Steuerungstechnik von Industrierobotern geben. Es wendet sich damit sowohl an Ingenieure, die mit der Entwicklung und dem Bau von Industrierobotern beschäftigt sind, als auch an den Anwender, der mehr über die innere Funktionsweise und das Zusammenspiel der Baugruppen des Steuerungssystems erfahren möchte.

Behandelt werden sowohl klassische Wissenszweige der Regelungs- und Steuerungstechnik, die für die Steuerung von Industrierobotern Bedeutung haben, als auch neuere Gebiete, die sich insbesondere unter dem Einfluß der Robotertechnik entwickelt haben. Dies betrifft vor allem die Sensortechnik und Sensorsignalverarbeitung und die Programmiersysteme für Industrieroboter. Moderne elektronische Robotersteuerungen arbeiten auf Mikrorechnerbasis. Deshalb wird besonderes Augenmerk auf die Darstellung von Steuerungs- und Verarbeitungsalgorithmen und ihre Realisierung mit den geräte- und programmtechnischen Mitteln der Mikrorechentechnik gelegt. Ein gesonderter Abschnitt ist der Programmierung von Industrierobotern gewidmet. Hier wird dargestellt, wie durch eine geeignete Softwaregestaltung die erforderliche Einsatzflexibilität erreicht werden kann. Bei der Anlage des Buches wurde unter Berücksichtigung des angesprochenen Leserkreises der Übersichtsdarstellung gegenüber einer ausführlichen Detailbehandlung der Vorzug eingeräumt. Die Gedankenlinie verläuft dabei von der Darstellung allgemeiner Grundlagen über die aus der Modellbildung der zu steuernden Komponenten des Industrieroboters abgeleiteten Verarbeitungsalgorithmen bis zur Beschreibung industrieller Robotersteuerungen.

Abschließend sei dem Verlag Technik für die großzügige Unterstützung und dem Lektor, Herrn *Reichenbach*, für die vertrauensvolle Zusammenarbeit herzlich gedankt.

Die Herausgeber

Inhaltsverzeichnis

1. Einführung 11

2. Grundprinzipien der Steuerung von Industrierobotern ... 13

2.1. Industrieroboter ... 13

2.2. Grundfunktionen einer Industrierobotersteuerung ... 17

2.3. Künstliche Intelligenz in Industrierobotersteuerungen 18

2.4. Grundstruktur einer Industrierobotersteuerung ... 20

2.4.1. Funktionssteuerung ... 21
2.4.2. Ablaufsteuerung und Befehlsarten 23
2.4.3. Dialogsteuerung und Betriebsarten ... 25

2.5. Einteilung von Industrierobotersteuerungen ... 28

2.6. Industrierobotersteuerungen auf Mikrorechnerbasis ... 29

3. Kinematische Systeme, Antriebe und Meßsysteme ... 33

3.1. Allgemeines ... 33

3.2. Funktionseinheiten des kinematischen Systems ... 34

3.2.1. Beispiele kinematischer Strukturen ... 34
3.2.2. Allgemeine Bewegung eines starren Kinematikglieds 35
3.2.3. Schubglied ... 36
3.2.4. Schwenkglied ... 37

3.3. Antriebe für das kinematische System ... 39

3.3.1. Elektrohydraulische Antriebe für Lageregelungen ... 40
3.3.1.1. Struktur der Antriebe und der Druckquelle ... 40
3.3.1.2. Bestandteile der Antriebe und ihre Modelle ... 41
3.3.1.3. Übertragungsverhalten und Kennwerte der Antriebe ... 48
3.3.1.4. Statisches und dynamisches Verhalten der Druckquelle ... 51
3.3.2. Elektrische Antriebe für Lageregelungen ... 53
3.3.2.1. Struktur ... 53
3.3.2.2. Bestandteile der elektrischen Antriebe und ihre Modelle ... 54
3.3.2.3. Übertragungsverhalten und Kennwerte des elektrischen Antriebs 59
3.3.3. In Lageregelungen eingesetzte pneumatische Antriebe 62
3.3.4. Elektrische Schrittantriebe ... 63
3.3.5. Einfachantriebe ... 63

3.4. Meßsysteme ... 64

3.4.1. Aufgaben ... 64
3.4.2. Probleme der Meßsignalgewinnung 65
3.4.3. Weg- und Winkelmessung ... 68
3.4.3.1. Allgemeine Probleme ... 68
3.4.3.2. Absolute Meßverfahren ... 70
3.4.3.3. Inkrementale Meßverfahren ... 71

3.4.3.4. Phasenzyklische Meßverfahren ... 73
3.4.4. Messung von Geschwindigkeit und Beschleunigung ... 75

4. Bewegungssteuerung ... 76

4.1. Grundprobleme ... 76

4.2. Modelle für das Grundgerät und die Antriebe ... 80
4.2.1. Allgemeine Ansätze ... 80
4.2.2. Modellierung mit Hilfe der Lagrangeschen Bewegungsgleichungen 2. Art ... 82
4.2.3. Modellierung über die allgemeine Bewegung starrer Glieder ... 88
4.2.4. Einfache Ersatzmodelle für Kinematik und Antriebe ... 90

4.3. Entkopplung der Achsen ... 93

4.4. Einzelachsregelung ... 98
4.4.1. Optimales Positionieren ... 98
4.4.2. Lineare Servoregler ... 103
4.4.3. Führungsverhalten des Antriebs ... 108
4.4.4. Beispiel einer digitalen Lageregelung ... 112

4.5. Adaptive Regelung ... 119

4.6. Bahnsteuerung ... 123
4.6.1. Bahnsteuerverfahren bei Multipunktsteuerungen ... 124
4.6.2. Koordinatentransformation ... 133
4.6.3. Homogene Transformationen ... 134
4.6.4. Direkte Koordinatentransformation ... 136
4.6.5. Inverse Koordinatentransformation ... 141
4.6.6. Realisierung der Koordinatentransformation ... 144
4.6.7. Trajektorienplanung und Interpolation ... 146

5. Aktionssteuerung ... 149

5.1. Grundfunktionen und Steuerungsentwurf ... 149
5.1.1. Grundfunktionen ... 149
5.1.2. Entwurfsstrategie ... 150
5.1.3. Steuerungsrealisierung 152

5.2. Mittel zur Funktionsbeschreibung ... 155
5.2.1. Überblick ... 155
5.2.2. Programmablaufgraphen 156
5.2.3. Netze ... 157

5.3. Modellbildung ... 166

5.4. Funktioneller Entwurf 167
5.4.1. Überblick ... 167
5.4.2. Entwurf mit Steuernetzen oder PAG ... 167
5.4.3. Entwurf mit Ablaufkontrollnetzen ... 177

5.5. Struktureller und technischer Entwurf ... 184
5.5.1. Ermittlung der logischen Gleichungen aus dem PAG ... 184
5.5.2. Ermittlung der logischen Gleichungen aus dem Steuernetz ... 186
5.5.3. Implementierung des AKN-Entwurfs mit dem Softwareentwurfssystem PSI 1520 187

6. Programmierung von Industrierobotern ... 195

6.1. Industrieroboter als programmierbare Systeme ... 195

6.2. Programmierverfahren für Industrieroboter ... 196
6.2.1. Manuelle Programmierung (Verbindungsprogrammierung) ... 196

6.2.2. Teach-in-Programmierung ... 198
6.2.3. Direktes Teach-in ... 198
6.2.4. Indirektes Teach-in ... 201
6.2.5. Textuelle Programmierung ... 204
6.2.6. Daten und Befehle im Roboterprogramm ... 204
6.2.7. Einfache Manipulatorsprachen ... 206
6.2.8. Höhere Programmiersprachen für Industrieroboter ... 214

6.3. Programmiersysteme von Industrierobotersteuerungen ... 216
6.3.1. Grundfunktionen eines Programmiersystems ... 216
6.3.2. Softwareentwicklung für Programmiersysteme ... 217
6.3.3. Modularer Aufbau von Steuerungssystemen ... 218
6.3.4. Hierarchische Gestaltung von Programmiersystemen ... 220

7. Sensoren ... 222

7.1. Äußere Sensoren und Steuerungsfunktionen 223

7.2. Taktile Sensoren ... 225
7.2.1. Taktile Meßfühler ... 225
7.2.2. Kraft-Moment-Sensoren ... 226
7.2.3. Taktile Matrixsensoren ... 232

7.3. Näherungssensoren ... 233
7.3.1. Anwesenheitssensoren ... 233
7.3.2. Distanzsensoren ... 235

7.4. Visuelle Erkennungssysteme ... 241
7.4.1. Bilderzeugung und Bildwandlung ... 242
7.4.2. Bildvorverarbeitung ... 248
7.4.3. Merkmalsgewinnung und Klassifikation ... 251
7.4.4. Visuelles System zur Teileerkennung und Lagebestimmung ... 258
7.4.5. 3-D-Erkennung ... 261

8. Industrielle Steuereinrichtungen ... 262

8.1. Allgemeines ... 262

8.2. Mikrorechnersystem K1520 ... 262

8.3. System 600 ... 266
8.3.1. Übersicht ... 266
8.3.2. Industrierobotersteuerung IRS 600 ... 269
8.3.2.1. Gerätetechnik ... 269
8.3.2.2. Programmsysteme und Programmierung ... 271
8.3.3. Industrierobotersteuerung IRS 650 ... 273
8.3.3.1. Gerätetechnik ... 275
8.3.3.2. Programmsysteme ... 276
8.3.3.3. Programmierung ... 277

8.4. Industrierobotersteuerung ursalog 5021 ... 278
8.4.1. Allgemeine Charakteristik ... 278
8.4.2. Bedienung und Programmierung ... 280
8.4.3. Funktionsbeschreibung ... 285

Anhang ... 295

A1. Numerisches Steuerungssystem CNC 600 ... 295

A1.1. Gerätetechnik des Logikschranks ... 297

A1.2. Komplex Leitrechner 298
A1.3. Bedien- und Anzeigekomplex 300
A1.4. Wegesteuerungs- und Lageregelungskomplex 302
A1.5. PMC-Komplex 302
A1.6. Rechnerkopplungen 303
A1.7. Prozeß-Ein-/Ausgabe-Steuerung 305
A1.8. Software des Steuerungssystem CNC 600 306
A1.9. Steuerprogrammsystem 306
A1.10. Serviceprogrammsystem 309
A1.11. Programmierung 311
A1.12. Freiprogrammierbare Steuerung für automatische Fertigungszellen CNC 603 314
A2. Gerätesystem ursalog 5000 315
A2.1. Speicherprogrammierbare Steuereinrichtung ursalog 5010 316
A2.2. Speicherprogrammierbare Steuereinrichtung ursalog 5020 318
A2.3. Programmsystem der speicherprogrammierbaren Steuereinrichtung ursalog 5020 322
A2.4. Fachsprache PROLOG 2 324

Literaturverzeichnis 335

Sachwörterverzeichnis 342

1. Einführung

Industrieroboter werden etwa seit Ende der sechziger Jahre industriell hergestellt und als universelle Automatisierungsmittel eingesetzt. Sie stellen eine wesentliche Voraussetzung für eine bedienerarme und flexible Gestaltung und Durchführung des Produktionsprozesses dar. Während der konventionelle Produktionsprozeß mit prozeßspezifischen Automatisierungsmitteln ausgerüstet ist, wird der Fortschritt, der durch die Industrierobotertechnik eintritt, durch zwei wesentliche Prinzipien getragen:

- den Einsatz von universell beweglichen Geräten, die zur Handhabung von Werkstücken und Werkzeugen und zur Durchführung technologischer Prozeßschritte eingesetzt werden können und in Serienproduktion rationell herstellbar sind;
- die Ausrüstung dieser Geräte mit einer programmierbaren Steuerung, die die erforderlichen Operationen automatisch auszuführen gestattet.

Die beiden Grundeigenschaften der leichten Anpaßbarkeit an unterschiedliche technologische Arbeitsplätze und der Programmierbarkeit der verschiedenen Operationsfolgen machen nicht nur den einzelnen Industrieroboter flexibel einsetzbar, sondern übertragen diese Flexibilität auch auf die roboterbestückte Fertigungszelle oder Fertigungslinie.

Industrieroboter konnten erst rationell gefertigt und effektiv eingesetzt werden, als es gelang, sie mit kompakten Steuerungssystemen auszurüsten, die die zweite der genannten Grundeigenschaften ausreichend gewährleisteten. Deshalb bildete die Mikroelektronik mit ihren rasch anwachsenden Verarbeitungsmöglichkeiten eine notwendige Voraussetzung für die Erfolge der Industrierobotertechnik, und es ist kein Zufall, daß der starke Aufwärtstrend der Industrieroboter nahezu zeitsynchron mit der Entwicklung und Vervollkommnung der Mikroprozessoren und Mikrorechner verläuft.

Einerseits ist der Industrieroboter für die Steuerungstechnik nicht in allen seinen Verhaltenseigenschaften ein völlig neues Steuerungsobjekt. Auf viele Erfahrungen aus der Steuerung von Werkzeugmaschinen konnte und kann bei der Entwicklung von Robotersteuerungen zurückgegriffen werden. Andererseits sind durch die speziellen Achsanordnungen, die relativ hohen Verfahrgeschwindigkeiten, die enge Kommunikation mit dem technologischen Prozeß und die besonderen Anforderungen an die Programmierung und Bedienung neue Prinzipien entwickelt und erforscht worden, die der Robotersteuerungstechnik ihr besonderes Gepräge geben. Diese Technik soll in diesem Buch zusammengefaßt dargestellt werden.

Im Abschnitt 2. werden die Grundprinzipien der Steuerung von Industrierobotern behandelt. Ausgehend von den Steuerungsaufgaben wird eine Strukturierung des Steuerungssystems vorgenommen und eine Übersicht über die Funktion der einzelnen Teilsysteme gegeben.

Abschnitt 3. behandelt Funktionseinheiten von Industrierobotern, soweit und wie sie für die Steuerungstechnik relevant sind. Das bezieht sich besonders auf die Antriebe und Meßsysteme, die direkt mit der Steuerung kommunizieren, aber auch auf das kinematische System, dessen Auslegung und Eigenschaften bei der Gestaltung der Bewegungssteuerung zu berücksichtigen sind. Die Bewegungssteuerung selbst wird in ihren beiden

Komponenten, der Bahnsteuerung und der Servoregelung, im Abschnitt 4. besprochen. Nach der Vorstellung der Grundprobleme erfolgt die Modellbildung für das kinematische System einschließlich der Antriebe und die Behandlung ausgewählter Regelungsverfahren. Der zweite Teil dieses Abschnitts ist dann der Bahnsteuerung zur Erzeugung der Sollwertverläufe für das Servoregelungssystem vorbehalten.

Abschnitt 5. enthält eine systematische Darstellung der binären Steuerungsaufgaben für einen Industrieroboter und deren Lösung. Beschrieben werden hier Methoden zur Ablaufplanung und zur Implementierung von Ablaufsteuerungen mit Mikrorechnern. Die Programmierung von Industrierobotern wird im Abschnitt 6. behandelt. Hier geht es sowohl um die Darstellung der einzelnen Programmierverfahren als auch um technische Realisierungsmethoden. Der textuellen Programmierung und den höheren Programmiersprachen wird dabei besondere Aufmerksamkeit gewidmet. Die Verarbeitung von Sensordaten in Robotersteuerungen ist Gegenstand von Abschnitt 7. Hier werden die grundsätzlichen Verarbeitungsschritte erläutert, die Ableitung von Steuerungsinformationen aus Sensordaten dargestellt und ausgewählte Beispiele besprochen.

Abschnitt 8. gibt einen zusammenfassenden Überblick über die gegenwärtig in der DDR verfügbare industrielle Realisierungsbasis für Industrierobotersteuerungen. Behandelt werden sowohl modulare Gerätesysteme, mit denen Steuerungen projektiert werden können, als auch Ausführungsbeispiele serienmäßiger Robotersteuerungen.

2. Grundprinzipien der Steuerung von Industrierobotern

2.1. Industrieroboter

Industrieroboter (IR) sind universell einsetzbare technologische Automatisierungsmittel, die durch folgende Merkmale gekennzeichnet sind:

- IR sind automatische Handhabungseinrichtungen für den industriellen Einsatz.
- Sie sind in mehreren Freiheitsgraden (Bewegungsachsen) im Raum beweglich.
- Die Operationen eines IR sind freiprogrammierbar.

Die Industrieroboter müssen von den Manipulatoren unterschieden werden. Manipulatoren sind Handhabungseinrichtungen, die mit Hilfe von Greifern und Greiferführungsgetrieben Hand und Arm des Menschen nachbilden und zum Bewegen von Gegenständen im Raum geeignet sind. Erfolgt die Steuerung unmittelbar per Hand, so spricht man von einem Synchronmanipulator oder Balancer.

Die automatische Arbeitsweise von Industrierobotern schafft die Voraussetzung für den Ersatz menschlicher Arbeitskräfte an den unterschiedlichsten Arbeitsplätzen. Die universelle Beweglichkeit gewährleistet dabei eine weitgehend beliebige Positionierung und Orientierung des am Roboterhandpunkt angebrachten Greifers oder Werkzeugs und ist damit eine notwendige Voraussetzung für die Erfüllung von Handhabe- und technologischen Aufgaben.

Die von einem Industrieroboter auszuführenden Operationen lassen sich in Bewegungen und Aktionen einteilen. Bewegungen werden durch gesteuertes Verfahren der Achsen unter Berücksichtigung numerischer Vorgaben (Positionen, Geschwindigkeiten) erzeugt. Aktionen sind vorwiegend binäre Operationen, die Schaltvorgänge am Handpunkt (z. B. Greifer öffnen, Farbstrahl freigeben) oder in der IR-Peripherie (z. B. Rundtisch weiterschalten) auslösen.

Freie Programmierbarkeit bedeutet schließlich, daß der Bediener/Programmierer zu beliebigen Zeitpunkten über eigens dafür vorgesehene Funktionselemente (Bedienfeld, Programmierhilfsgerät) eine beliebige Folge von Operations- (Bewegungs- und Aktions-) Befehlen eingeben und bei Bedarf ändern kann. Das Gegenstück dazu sind festprogrammierte Systeme, bei denen in der Inbetriebnahmephase das Programm erstellt und eingefahren wird und später nur durch gerätetechnische Eingriffe in das System (z. B. Austausch des Programmspeichers, mechanische Veränderung von Schaltnocken) geändert werden kann.

Der Einsatz eines oder mehrerer Industrieroboter ist eine technologische Aufgabe, die sich in die Teilprobleme

- Gestaltung des Roboterarbeitsplatzes und der Peripherie des Industrieroboters (unter letzterem wird die Gesamtheit aller technologischen Hilfseinrichtungen, z. B. Spannvorrichtungen, Ablagen, Paletten und Schutzvorrichtungen, verstanden, die der IR zur Erfüllung seiner technologischen Aufgabe benötigt);
- Auswahl bzw. Entwicklung des Grundgeräts (darunter wird der maschinenbautechni-

sche Teil des IR einschließlich der Antriebe, der Meßwertaufnehmer und der Stellglieder verstanden);
- Auswahl bzw. Entwicklung der IR-Steuerung.

Aufgrund ihrer engen Wechselbeziehungen können diese drei Teilaufgaben nur gemeinsam im engen Zusammenwirken von Technologen, Maschinenbauern und Steuerungstechnikern erfolgreich gelöst werden. Einige dieser Wechselbeziehungen, die insbesondere den Steuerungstechniker interessieren, sind im Bild 2.1 angegeben. Zunächst stellen aufgrund der technologischen Aufgabe das Grundgerät und die Peripherie Anforderungen an die Steuerung, die insbesondere
- die Art der Bewegungssteuerung,
- die Anzahl der zu steuernden Achsen und der binären Steuersignale,
- die Kommunikation mit anderen an der Lösung der technologischen Aufgabe beteiligten Einrichtungen (z. B. Bearbeitungsmaschinen und Fördereinrichtungen),
- die Überwachung und Gewährleistung der Prozeßsicherheit

betreffen.

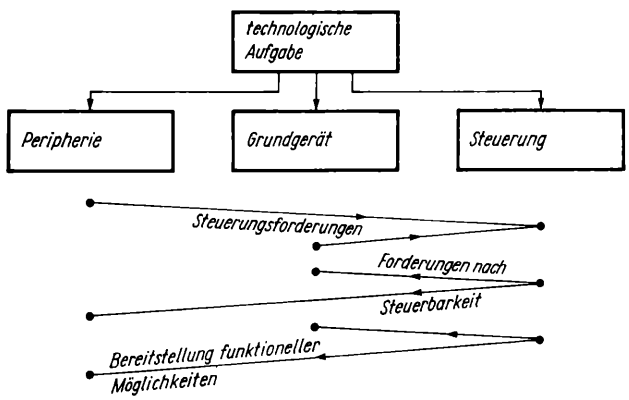


Bild 2.1
Teilprobleme bei einem Industrierobotereinsatz

Außerdem werden an die zu steuernden Objekte Steuerbarkeitsforderungen gestellt. Sie beziehen sich u. a. auf
- die Auslegung des Grundgeräts (z. B. Spielfreiheit des Signalwegs Antrieb–Meßsysteme,
- die Festlegung des Charakters und der Art der auszutauschenden Steuersignale (z. B. Signalpegel, parallele oder serielle Übertragung),
- Schutzmaßnahmen gegen Störeinflüsse (z. B. galvanische Trennung von Steuerung und Steuerungsobjekt).

Schließlich ergeben sich bei der Lösung der technologischen Aufgabe aus steuerungstechnischer Sicht häufig neue funktionelle Möglichkeiten, die über die ursprüngliche Aufgabenstellung hinausgehen. Sie beziehen sich vorwiegend auf die Erhöhung von Qualität und Quantität der steuerungsinternen Verarbeitungsleistung und bewirken eine verbesserte Prozeßanpassung und bessere Eigenschaften oder erweiterte Möglichkeiten der Bedienung und Programmierung des Roboters.

Bezüglich des Einsatzes lassen sich drei Klassen von IR unterscheiden (Bild 2.2):

Die *Beschickungsroboter* führen technologische Hilfsoperationen (vorwiegend Beschicken von Werkzeugmaschinen, aber auch andere Transportoperationen) aus. Sie sind dazu mit einem Greifer ausgerüstet, kommunizieren relativ stark mit der den technologischen Prozeß ausführenden Einrichtung und werden mitunter auch von der dieser Einrichtung (Werkzeugmaschine) zugeordneten Steuerung mit gesteuert. Als Bewegungs-

verhalten ist das sequentielle Anfahren diskreter Raumpunkte ohne funktionellen Zusammenhang der Achsbewegung (PTP Punkt-zu-Punkt-Steuerung) typisch. In einfachen Fällen, in denen mit festen Anschlägen gearbeitet werden kann, genügt eine binäre Ablaufsteuerung; bei einer größeren Anzahl anzufahrender Raumpositionen werden Lagerregelkreise mit analog oder digital vorgebbaren Sollwerten eingesetzt. Im letzteren Fall spricht man auch von numerisch gesteuerten Achsen.

Der *technologische Roboter* (Bild 2.3) führt technologische Hauptoperationen, wie Schweißen, Farbspritzen, Schleifen und Entgraten, selbständig aus. Er trägt am Hand-

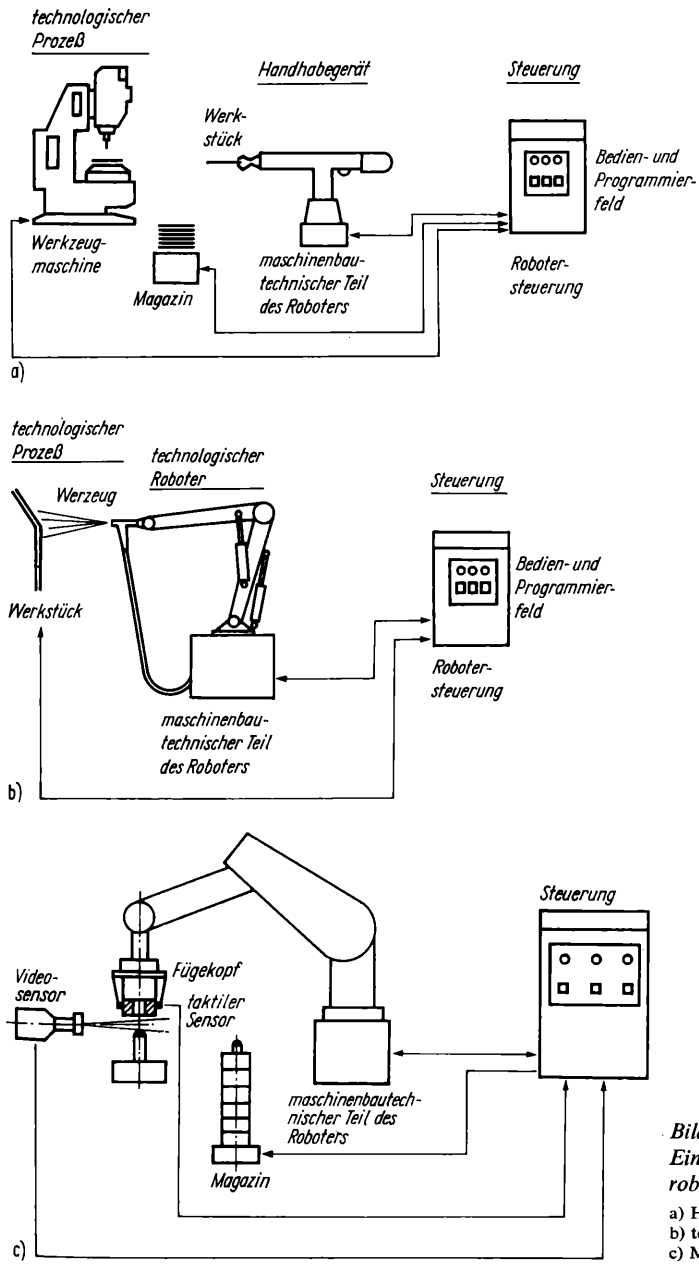


Bild 2.2
Einsatzklassen von Industrierobotern

a) Handhabegerät (nach [2.2])
b) technologischer Roboter
c) Montageroboter

punkt entweder ein Werkzeug (z. B. Schweißpistole, Schleifmaschine) oder einen Greifer, mit dem er das Werkstück an die Arbeitsmaschinen heranführt und bearbeitet. Er verfügt über eine Steuerung, die das Programmieren und Reproduzieren von Bewegungsbahnen in mehreren Achsen gestattet (Bahnsteuerung). Die Antriebe sind dazu fast immer lagegeregelt. Eine Ausnahme stellen IR zum Punktschweißen dar, die mit einer Punkt-zu-Punkt-Steuerung auskommen.

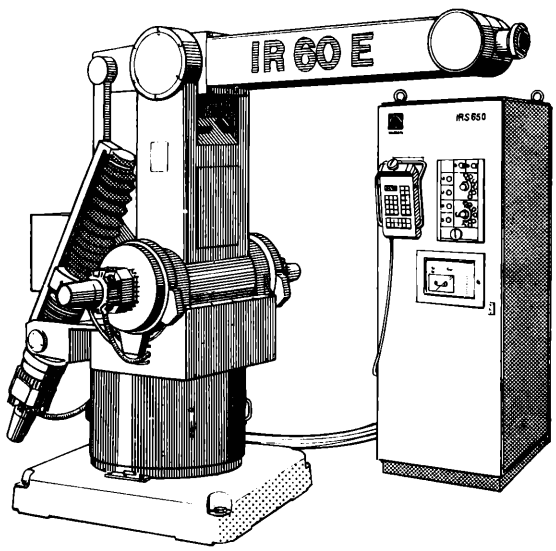


Bild 2.3
Technologischer Roboter
(mit freundlicher Genehmigung des VEB Kombinat Fortschritt)

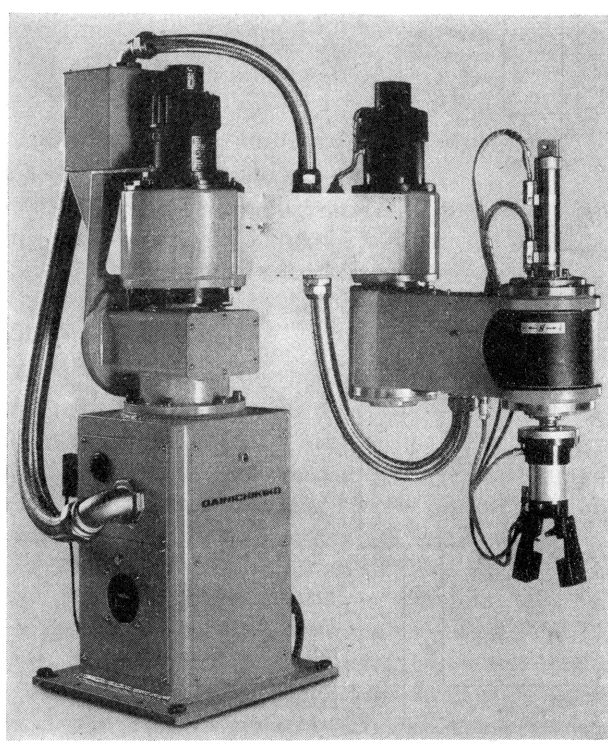


Bild 2.4
Montageroboter
(mit freundlicher Genehmigung der Fa. Stiefelmayer)

Der *Montageroboter* (Bild 2.4) führt automatisch Montageoperationen aus. Er ist dazu mit einem Montagegreifer (Fügekopf) ausgerüstet. Die Steuerung gewährleistet in der Regel sowohl Punkt-zu-Punkt- als auch Bahnbewegungen und muß diese Bewegungen auch unter Berücksichtigung von Zusatzbedingungen (z. B. begrenzte Fügekraft) ausführen. Montageroboter kommunizieren sehr stark mit der technologischen Umwelt und sind gewöhnlich mit einem umfangreichen System von Sensoren ausgerüstet.

Die Unterscheidung dieser drei Einsatzklassen bedeutet nicht, daß alle Industrieroboter jeweils auf eine dieser Klassen zugeschnitten sind. So gibt es einerseits sehr viele Geräte, die ausschließlich in der Werkstückhandhabung eingesetzt werden können; andererseits gestatten universellere Industrieroboter einen Einsatz sowohl in der Handhabung wie auch als technologische Roboter oder sogar in allen drei der genannten Einsatzklassen. Ob für einen bestimmten Einsatzfall ein universelles oder ein entsprechend zugeschnittenes Gerät auszuwählen ist, wird vorwiegend durch ökonomische und produktionsorganisatorische Bedingungen bestimmt.

2.2. Grundfunktionen einer Industrierobotersteuerung

Wesentliche Robotereigenschaften, wie Programmierbarkeit, Anpassungsfähigkeit und Einsatzflexibilität, werden durch die Steuerung gewährleistet oder entscheidend beeinflußt. Im wesentlichen erfüllt eine IR-Steuerung folgende Aufgaben, die insgesamt auch die Steuerungsstruktur prägen:

- Realisierung des Gesamtablaufs entsprechend dem eingegebenen Programm, dafür sorgt eine Funktionsgruppe *Ablaufsteuerung*;

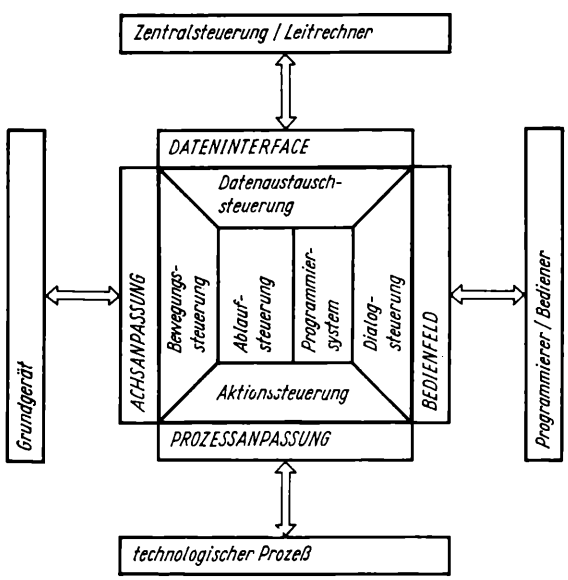


Bild 2.5
Schnittstellen und zugeordnete Funktionsgruppen einer IR-Steuerung

- Gewährleistung der Programmierbarkeit des Industrieroboters, dafür ist eine Funktionsgruppe *Programmiersystem* vorgesehen, die die Ein- und Ausgabe, den Test und die Änderung von Roboterprogrammen ermöglicht;
- Steuerung, Koordinierung und Überwachung der Bewegungen des Grundgeräts, dafür ist die Funktionsgruppe *Bewegungssteuerung* zuständig;

- Kommunikation mit der technologischen Umwelt zur Koordinierung von Aktions- und Bewegungsabläufen und Synchronisation mit dem technologischen Prozeß, sie erfolgt in der Funktionsgruppe *Aktionssteuerung* (gelegentlich auch als Technologiesteuerung bezeichnet);
- Kommunikation mit dem Bediener/Programmierer und dem Wartungs- bzw. Instandhaltungspersonal, sie wird über eine *Dialogsteuerung* abgewickelt;
- Kommunikation mit anderen (IR- oder Werkzeugmaschinen-) Steuerungen oder mit einem Leitrechner, dafür ist die Funktionsgruppe *Datenaustauschsteuerung* vorgesehen.

Bild 2.5 gibt eine Übersicht über die Schnittstellen einer IR-Steuerung und die jeweils angeschlossenen Funktionsgruppen. Je nach Einsatzfall und dadurch bedingten Ausrüstungsgrad der Steuerung können einzelne Funktionsgruppen mehr oder weniger stark ausgebaut sein oder auch ganz fehlen. So besteht beispielsweise für ein einfaches Handhabungsgerät mit numerisch gesteuerten Achsen die Bewegungssteuerung einfach in der sequentiellen Aufschaltung des jeweils programmierten Positionssollwerts auf die Lagerregelkreise und einer Rückmeldung nach Erreichen dieser Position, während bei einem Montageroboter die Aktionssteuerung bzw. die Prozeßanpassung (Bild 2.5) durch ein umfangreiches System zur Bildauswertung ergänzt werden kann. Das Programmiersystem ist um so umfangreicher, je komfortabler die Möglichkeiten der Programmerstellung und des Programmtests bei einem Industrieroboter sind. Es ist in elementarer Form bei teach-in-programmierbaren Robotern (s. 6.2.2.) und in weit ausgebauter Form beim Einsatz von problemorientierten Robotersprachen vorhanden.

Manuell programmierte Handhabungsgeräte benötigen kein Programmiersystem. Hier übernimmt der Mensch durch manuelle Operationen vollständig die Programmerstellung und -änderung.

2.3. Künstliche Intelligenz in Industrierobotersteuerungen

Begriffe wie „künstliche Intelligenz" oder „intelligente Geräte" werden immer häufiger als Kennzeichen für ein hohes qualitatives Niveau der automatischen Lösung von Problemen und einen großen Umfang an installierter Verarbeitungsleistung benutzt. Eine allen Ansprüchen gerecht werdende Definition der künstlichen (in Abgrenzung zur menschlichen) Intelligenz ist bisher noch nicht gelungen. Hier sollen einige Aspekte dieser Problematik kurz dargestellt werden, die mit dem Ersatz menschlicher Arbeitsleistung durch Maschinenleistung im Zusammenhang stehen und von Bedeutung für die Realisierung von Robotersteuerungen sind. Bei der Lösung einer technologischen Aufgabe werden im Prinzip immer drei hierarchisch aufeinander aufbauende Operationsstufen durchlaufen (Bild 2.6) [2.12].

Ausgehend von der allgemeinen Zielstellung wird zunächst ein Lösungskonzept erarbeitet und daraus eine Planung der Lösung in Form von Teilproblemen abgeleitet (strategisches Niveau). Anschließend erfolgt eine Algorithmierung der Teilprobleme in Form der Festlegung von Folgen aus komplexen Operationen (taktisches Niveau). Schließlich werden auf operativem Niveau die komplexen Operationen in Elementaroperationen aufgelöst.

Die künstliche Intelligenz oder, hier synonym, das Niveau der Verarbeitungsleistung eines Industrieroboters kann nun danach bewertet werden, welche der vorgenannten Aufgaben durch die Robotersteuerung übernommen werden und welche folglich noch durch den Bediener/Programmierer gelöst werden müssen. Bei einem direktgesteuerten Mani-

pulator liegen alle drei Verarbeitungsniveaus beim Bediener, der das Gerät über Stell- und Schaltsignale steuert. Ein programmierbarer Industrieroboter kann Folgen von Bewegungs- und Aktionsbefehlen ausführen, wobei die konzeptive Planung und Algorithmierung der Problemlösung dem Bediener überlassen sind. Ein auf taktischem Niveau arbeitendes Steuerungssystem kann komplexe Instruktionen verarbeiten, während ein implementiertes strategisches Programmiersystem eine globale Programmierung durch Zielvorgaben gestattet. Die algorithmische Auflösung in Teilschritte wird dann durch die Steuerung selbst vorgenommen. Im Bild 2.6 rechts sind dazu Beispiele für Anweisungen auf jedem Niveau in umgangssprachlicher Notierung angegeben. Kleingeschrieben und unterstrichen sind dabei Anweisungen; großgeschrieben sind vereinbarte Variable oder Parameter.

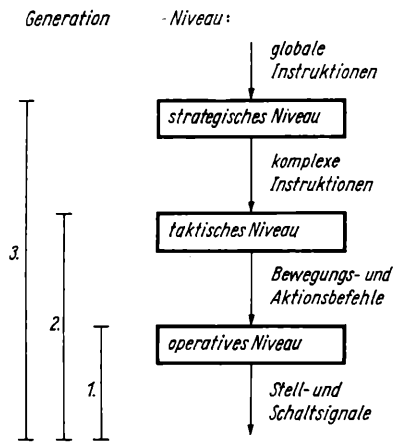


Bild 2.6 Hierarchie von Verarbeitungsniveaus

Mit steigendem Verarbeitungsniveau bedarf die Steuerung immer umfassenderer Informationen über die technologische Umwelt des Industrieroboters. Die Gesamtheit aller in der Steuerung vorhandenen Informationen dieser Art wird als Umweltmodell bezeichnet. Im einfachsten Fall (operatives Niveau) wird die Umwelt bei der Auslegung der Steuerung und bei der Programmierung durch den Menschen fest installiert. Roboter mit taktischem Verarbeitungsniveau bedürfen bereits ständiger Informationen über den aktuellen Zustand der Umwelt. Letzterer wird durch Steuersignale erfaßt. Strategische Operationen erfordern ein komplexes Umweltmodell und eine Bewertung im Sinne der vorgegebenen Zielstellung. So sind die besprochenen Verarbeitungsniveaus gleichzeitig für die Einteilung von Industrierobotern nach Generationen charakteristisch (Bild 2.6 links). Tafel 2.1 gibt eine Übersicht über die gegenwärtig definierten drei Robotergenerationen mit ihren wesentlichen Merkmalen.

Im Unterschied zu der bekannten Einteilung von Digitalrechnern in Generationen, die sich durch ihre Realisierungsbasis (Integrationsgrad und Art der eingesetzten Bauelemente) unterscheiden, erfolgt die Klassifizierung der Robotergenerationen nach ihrem Funktionsinhalt. Dabei nimmt die Leistungsfähigkeit mit steigender Generation zu. Damit ist der Begriff „Generationen" nicht als zeitliche Aufeinanderfolge von Gerätesystemen aufzufassen, die einander ablösen, sondern im Sinne einer sich ständig erweiternden Breite des IR-Einsatzfelds. Für Anwendungsfälle, bei denen der Funktionsinhalt eines Roboters der ersten Generation bereits hinreicht, ist somit auch ein Generationswechsel im obigen Sinne nicht zu erwarten und nicht notwendig.

Tafel 2.1. Generationen von Industrierobotern

	Mensch-Maschine-Interface (Programmierung)	Interface zum technologischen Prozeß	Realisierung, typische Baugruppen
1. Generation programmierbare IR	manuelle Programmierung (Steckerfeld, Verkabelung, PROM) Teach-in (direkt und indirekt) Aktions- und Programmablaufbefehle in maschinenorientierter Programmiersprache	einfache Kontroll- und Überwachungselemente mit binärem Ausgang binäre Signale zur Koordinierung von Synchronisation von Bewegungen und Aktionen Reaktion auf Umweltsituationen durch bedingte Befehle oder durch Stillsetzen des IR Steuerung vorwiegend autonom arbeitende oder in der Steuerung der zu bedienenden Einheit integrierte Systeme	konventionelle Elektronik Mikrorechnerbaugruppen als Alternative zu Hardwarelösungen
2. Generation sensitive IR	erweiterte Teach-in-Verfahren (automatische Suche der Bewegungsbahn, Servounterstützung) systemorientierte algorithmische Programiersprachen (ablauforientierte Programmierung)	Sensorsysteme zur Erfassung technologischer Teilsituationen binäre und analoge vektorielle Signale zur Reaktion auf Umweltveränderungen Reaktion durch spezielle Unterprogramme (ausgebautes Interruptsystem)	Realisierung auf der Basis von Minirechnern oder Mikrorechnersystemen komplexe, auch mehrdimensionale Sensorsysteme Steuerungen untereinander und mit einer Zentralsteuerung kopplungsfähig weitgehend modularer Aufbau
3. Generation intelligente IR	problemorientierte Programmiersprachen mit der Möglichkeit der Einbeziehung von Teach-in-Programmelementen globale (zielorientierte) Programmierung mit komplexem Umweltmodell	intelligente Sensorsysteme zur komplexen Erfassung der technologischen Umwelt autonome Entscheidungsfindung und selbständige Planung von Teilabläufen	Rechnerverbundsysteme komplexe, mehrdimensionale Sensoren mit umfangreicher Signalvorverarbeitung Integrationsfähigkeit in CAM-Systeme prinzipiell modularer Aufbau

2.4. Grundstruktur einer Industrierobotersteuerung

Die aus den Grundaufgaben abgeleiteten Funktionsgruppen (s. Abschn. 1.2.) werden hierarchisch in unterschiedliche Steuerebenen eingeordnet (Bild 2.7) [2.1].

Der IR-Steuerung übergeordnet kann eine Zentralsteuerung (Leitrechner) sein. Sie ist immer dann erforderlich, wenn mehrere Industrieroboter zusammenarbeiten und ihre Bewegungen und Aktionen koordiniert werden müssen, und stellt in der Regel eine eigenständige Einheit mit einem zentralen Terminal dar, von dem aus eine komplette Fertigungszelle überwacht und gesteuert wird. Der Leitrechner kommuniziert über Datenschnittstellen mit den einzelnen Robotersteuerungen der Zelle und gibt insbesonders vor, welches Programm wann abzuarbeiten ist.

Die anderen im Bild 2.7 dargestellten Steuerebenen sind in die eigentliche IR-Steuerung integriert.

Die *Ablaufsteuerung* sorgt für die Abarbeitung des Roboterprogramms. Bei Bewegungs- und Aktionsbefehlen aktiviert sie die *Funktionssteuerung*, in der die Bewegungs- und Aktionssteuerung zusammengefaßt sind. Die letzte Stellebene, die *Effektorsteuerung*,

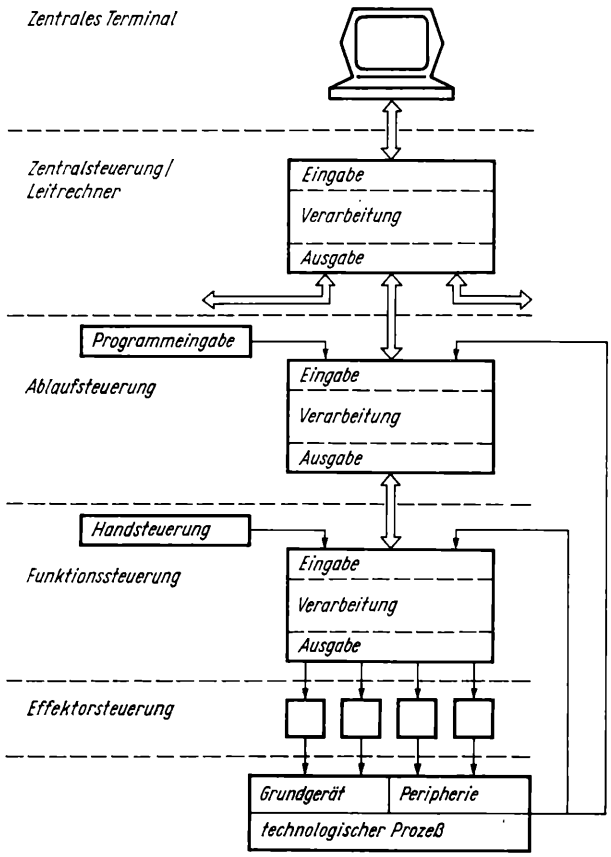


Bild 2.7
Hierarchische Struktur einer Robotersteuerung (nach [2.1])

besteht im wesentlichen aus den Leistungsverstärkern für die Antriebe und andere Stell- und Schaltglieder, die auf Grundgerät und Peripherie einwirken. Wesentlich für die einwandfreie Funktion des Steuerungssystems sind Rückmeldungen vom Grundgerät oder von der Peripherie. Davon sind im Bild 2.7 die wichtigsten eingezeichnet, die entweder auf die Funktionssteuerung (z.B. Positions-Istwerte, Fertigmeldungen) oder in die Ablaufsteuerung (Signale, die Bedingungen für den Programmablauf darstellen) zurückwirken. Funktions- und Ablaufsteuerung sollen nachfolgend genauer behandelt werden.

2.4.1. Funktionssteuerung

Die Funktionssteuerung ist der Ablaufsteuerung hierarchisch untergeordnet und enthält prinzipiell zwei Wirkungslinien:

Die *Bewegungssteuerung* (Bild 2.8 links) wertet die Bewegungsbefehle des Roboterprogramms aus und steuert in Abhängigkeit davon die Antriebe der einzelnen Achsen an. Bei einfachen Handhabegeräten mit Punkt-zu-Punkt-Steuerung schaltet sie in Abhängig-

keit von dem vorgegebenen Endpunkt und den Endschaltersignalen die Achsentriebe ein bzw. aus; bei Industrierobotern mit Bahnsteuerung besteht sie im Prinzip aus einem Bahnkurvenrechner (Interpolator), der in Echtzeit die Weg-Zeit-Verläufe der geforderten Bewegung ermittelt und als Sollwerte an die Lageregelung ausgibt. Bei hochwertigen Steuerungen kommt noch ein Funktionsblock für die Koordinatenwandlung hinzu. In der Bewegungssteuerung konzentriert sich in der Regel die numerische Verarbeitung des IR-Steuerungssystems.

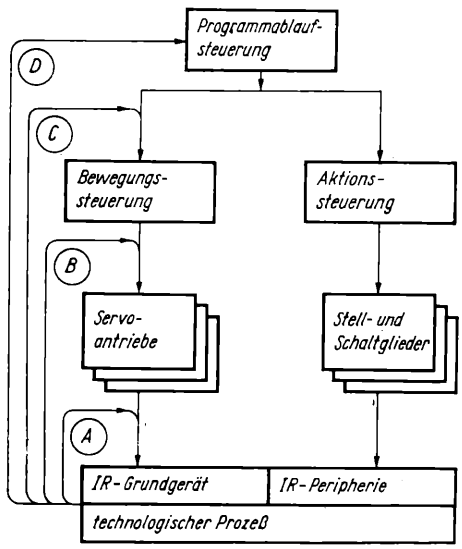


Bild 2.8
Funktionssteuerung eines Industrieroboters
(nach [2.4])

Die *Aktionssteuerung* (Bild 2.8 rechts) wertet die Aktionsbefehle des Roboterprogramms aus und steuert in Abhängigkeit davon die technologischen Aktionen des Roboters und der Peripherie über entsprechende Effektoren. Da die Aktionsbefehle eines Roboters in der Mehrzahl zweiwertige Wirkungen hervorrufen (z.B. Greifer auf–zu, Farbstrahl ein–aus, Rundtisch weiterschalten) und auch die zu ihrer Überwachung eingesetzten Sensoren (gegenwärtig noch) überwiegend binäre Signale abgeben, herrscht in der Aktionssteuerung die nichtnumerische (binäre) Verarbeitung vor. Beispiele für numerische Steuerungsprobleme in dieser Wirkungslinie sind die Regelung des Schweißstroms bei Lichtbogenschweißrobotern und die Regelung der Intensität des Farbstrahls bei Farbspritzrobotern entsprechend den numerischen Vorgaben im Roboterprogramm.

Das Niveau der in der Funktionssteuerung installierten Informationsverarbeitung zeigt sich in der Adaptivität des Roboters, d.h. in seiner Fähigkeit, auf Situationsänderungen in der technologischen Umwelt zu reagieren. Erreicht wird dies durch Informationsrückführungen auf verschiedenen Verarbeitungsniveaus. Im Bild 2.8 sind ganz links einige Möglichkeiten solcher Rückführungen angegeben. In Richtung zunehmender Adaptivität sind zu unterscheiden:

A. Rückführungen ohne Verarbeitung in der Steuerung

Sie werden durch geeignet nachgebende mechanische Konstruktionselemente erreicht. Ein typisches Beispiel ist das RCC-Glied (RCC remote-center of compliance), ein mechanischer Fügekopf, der das Verklemmen beim Fügen von Bolzen in Bohrungen durch Verlagerung des Momentanpols der Drehbewegung mit Hilfe nachgiebiger Elemente vermeidet und in der Montage Einsatz findet.

B. Rückführungen auf der Ebene der Servoantriebe

Damit sind quantitative koordinatenspezifische Korrekturen der Bewegungsbahn in Abhängigkeit von Sensorsignalen möglich. Das wird im einfachsten Fall durch Aufschaltung von Korrekturwerten auf die Sollwerte für die Lageregelung realisiert. Ein typischer Anwendungsfall sind Bahnkorrekturen beim Lichtbogenschweißen, wenn durch die Schweißteiltoleranzen die zu schweißende Naht von der einprogrammierten Bahn abweicht und diese Abweichung in einer Achse durch ein Nachführsystem mit Näherungssensor ausgeglichen wird.

C. Rückführungen auf der Ebene der Geometriesteuerung

Hier können quantitative und qualitative Bahnänderungen ausgeführt werden. Dazu werden in der Regel Bahnelemente, die nicht im Programm vorgesehen sind, selbsttätig in die programmierte Bewegungsbahn eingefügt. Ein typischer Anwendungsfall ist das automatische Umfahren von Hindernissen nach deren Erkennung durch geeignete Sensorsysteme. Auch die programmtechnische Einschränkung des Arbeitsbereichs von Industrierobotern läßt sich funktionell hier einordnen. Sie erfolgt dadurch, daß die programmierte Bahn ständig darauf überprüft wird, ob sie die Grenzen des einprogrammierten Bewegungsfreiraums überschreitet, und in Abhängigkeit von dieser Prüfung als zulässig erkannt und verfahren oder durch eine andere Bewegungsbahn an den Freiraumgrenzen ersetzt wird.

D. Rückführung auf der Ebene der Programmsteuerung

Sie gestattet eine selbständige Bewegungs- und Aktionsplanung auf der Basis einer komplexen Erfassung des Zustands der technologischen Roboterumwelt, die mit einem in der Steuerung programmierten Umweltmodell verglichen wird. Roboter mit solchen Steuerungen gestatten eine zielorientierte Programmierung (im Unterschied zur algorithmisch orientierten oder „expliziten" Programmierung aller anderen Typen).

Rückführungen der Art A werden bereits bei Robotern der ersten Generation eingesetzt, solche der Arten B und C sind für die zweite, solche der Art D für die dritte Generation typisch.

2.4.2. Ablaufsteuerung und Befehlsarten

Die Ablaufsteuerung realisiert den Gesamtablauf von Bewegungen und Aktionen gemäß dem eingegebenen Programm. Sie greift auf den Programmspeicher zu und sorgt für die sequentielle Ausführung der dort abgelegten Befehle.

Einfache Handhabegeräte kommen mit einer Ablaufsteuerung mit festem Programmablauf aus (Bild 2.9a). Die Ablaufsteuerung reduziert sich auf ein Schrittschaltwerk, das bei Eintreffen gewisser Quittungssignale aus Roboter oder Peripherie (z. B. Erreichen von Endlagenschaltern) inkrementiert wird, worauf der nächste Befehl der Funktionssteuerung zur Ausführung übergeben wird. Realisierungsbeispiele sind Kugelschrittschaltwerke und über Schrittschaltwerk angesteuerte Steckerfelder.

Bereits das einfache Beispiel im Bild 2.9a zeigt, daß aus dieser Sicht eine Industrierobotersteuerung als programmierbares System im Prinzip die Grundstruktur eines Rechners besitzt: sie verfügt über einen Programmspeicher; Ablaufsteuerung und Funktionssteuerung können zu einer verarbeitenden Einheit („Prozessor") zusammengefaßt werden, und es gibt schließlich (im Bild nicht genau dargestellt) ein Eingabe-/Ausgabewerk zur Kommunikation mit, von der Steuerung aus gesehen, peripheren Einrichtungen.

Bei festem Programmablauf werden alle Befehle von der Funktionssteuerung ausgeführt, die so dem Operationswerk (ALU) eines Rechners entspricht. Eine Ausnahme bildet im Bild 2.9a lediglich der Programmendebefehl (END), der als Organisationsbefehl auf die Ablaufsteuerung wirkt.

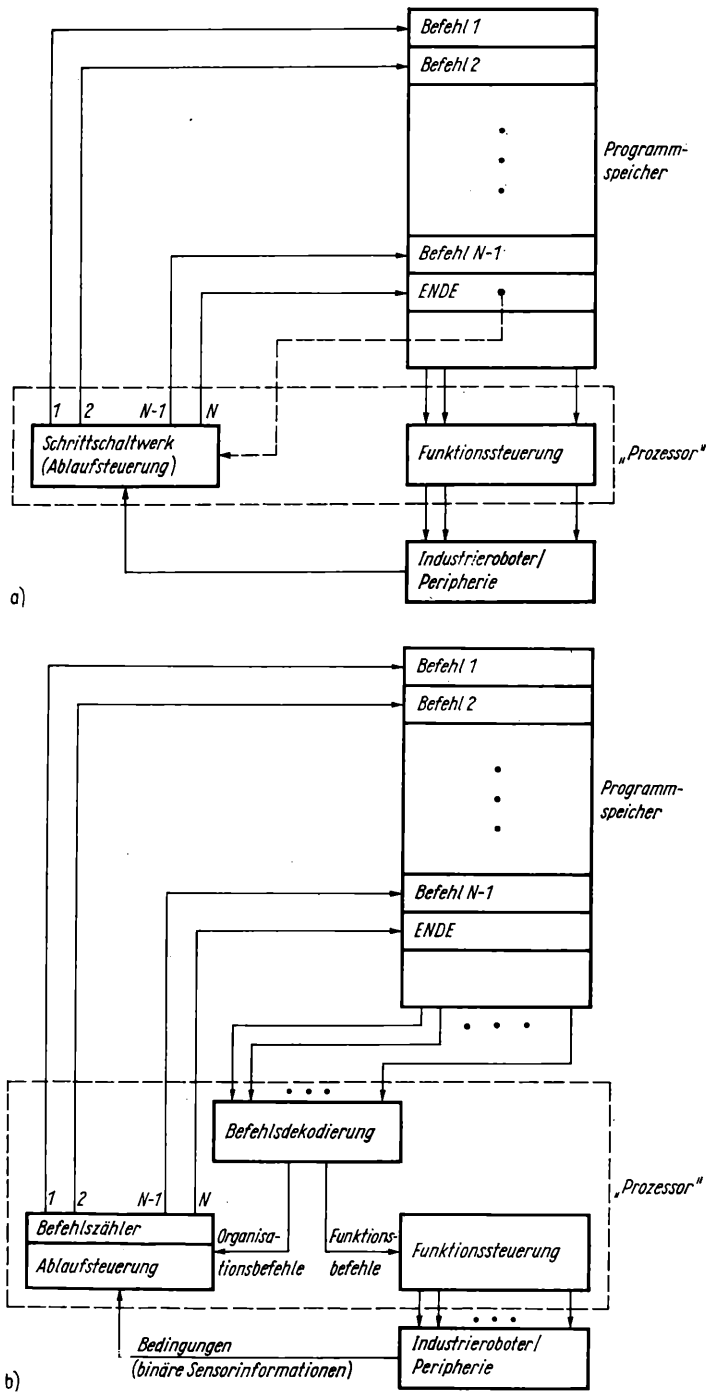


Bild 2.9
Ablaufsteuerung (nach [2.1])
a) mit festem Programmablauf
b) mit programmierbarem Programmablauf

Werden weitere Organisationsbefehle und die Möglichkeit ihrer Verarbeitung vorgesehen, so kommt man zur Ablaufsteuerung mit programmierbarem Programmablauf (Bild 2.9b). Hier muß vor Ausführung jedes Befehls geprüft werden, ob er als funktioneller Befehl der Funktionssteuerung oder als organisatorischer Befehl der Ablaufsteuerung zur Ausführung zugeleitet wird. Dafür ist eine Funktionsgruppe „Dekodierung" innerhalb des Prozesses zuständig. Funktionelle Befehle lösen eine Bewegung oder Aktion des Industrieroboters unmittelbar aus. Typische Beispiele sind:

Bewegungsbefehle
Bewege Achse X bis zum Endpunkt.
Fahre auf einer Geraden vom aktuellen Raumpunkt bis zum Punkt (X, Y, Z).

Aktionsbefehle
Greifer auf.
Schalte Steuerleitung Nr. N ein.

Wartebefehle
Setze den Programmablauf nach n Sekunden fort.
Warte, bis Steuerleitung Nr. N eingeschaltet wird.

Organisatorische Befehle wirken auf den Programmablauf. Ihre Ausführung kann von Bedingungen (Erreichen vorgegebener Zustände des Gesamtsystems) abhängig sein. Zu ihrer Ausführung verfügt die Ablaufsteuerung über einen einstellbaren Befehlszähler und Funktionsgruppen, die in Abhängigkeit vom auszuführenden Befehl und den erfüllten Bedingungen entscheiden, welcher Adreßwert in den Befehlszähler eingetragen und welcher Befehl damit als nächster Befehl ausgeführt wird.

Typische Beispiele für organisatorische Befehle sind:

Sprungbefehle
Setze das Programm bei Befehl Nr. A fort.
Setze das Programm bei Befehl Nr. B fort, wenn Signalleitung Nr. N eingeschaltet ist.

Unterprogrammaufrufbefehle und Rückkehrbefehle
Rufe das Unterprogramm mit Namen U auf.
Kehre ins Hauptprogramm zurück.

Die grundsätzliche Befehlsabarbeitung in einem programmierbaren System nach Bild 2.7 ist im Bild 2.10 dargestellt.

Sie gliedert sich in die Phasen

Befehlsaufstellung, bei der der Befehl in die verarbeitende Einheit (Prozessor) geholt wird, und

Befehlsdekodierung und -ausführung, bei der der Befehl zuerst entschlüsselt und anschließend an die betroffene Funktionsgruppe zur Ausführung weitergeleitet wird.

Die Befehlsaufstellung ist immer eine Transportoperation aus dem Programmspeicher in den Prozessor; die Dekodierung wird im Prinzip durch eine Mehrfachverzweigung realisiert, und die Ausführung besteht in der Regel in einer Folge von numerischen und nichtnumerischen Operationen in der Programmablauf- oder in der Funktionssteuerung.

2.4.3. Dialogsteuerung und Betriebsarten

Im Bild 2.11 ist die prinzipielle Gesamtstruktur der Steuerung eines autonom arbeitenden Industrieroboters zusammenfassend dargestellt. Die wesentlichsten Informationsflüsse zwischen den Funktionseinheiten sind eingetragen. Deutlich zu erkennen sind die Hier-

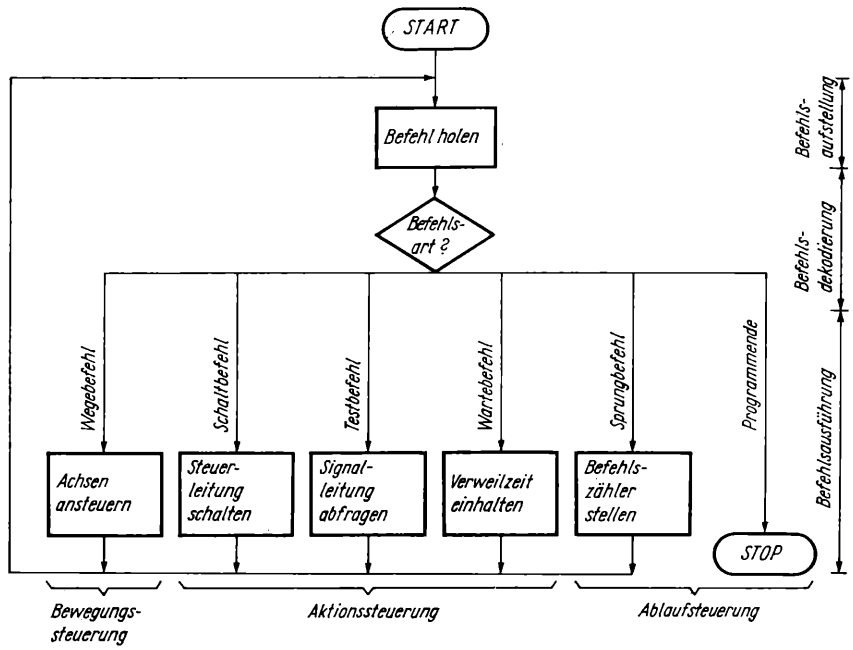


Bild 2.10. Befehlsabarbeitung in einer IR-Steuerung

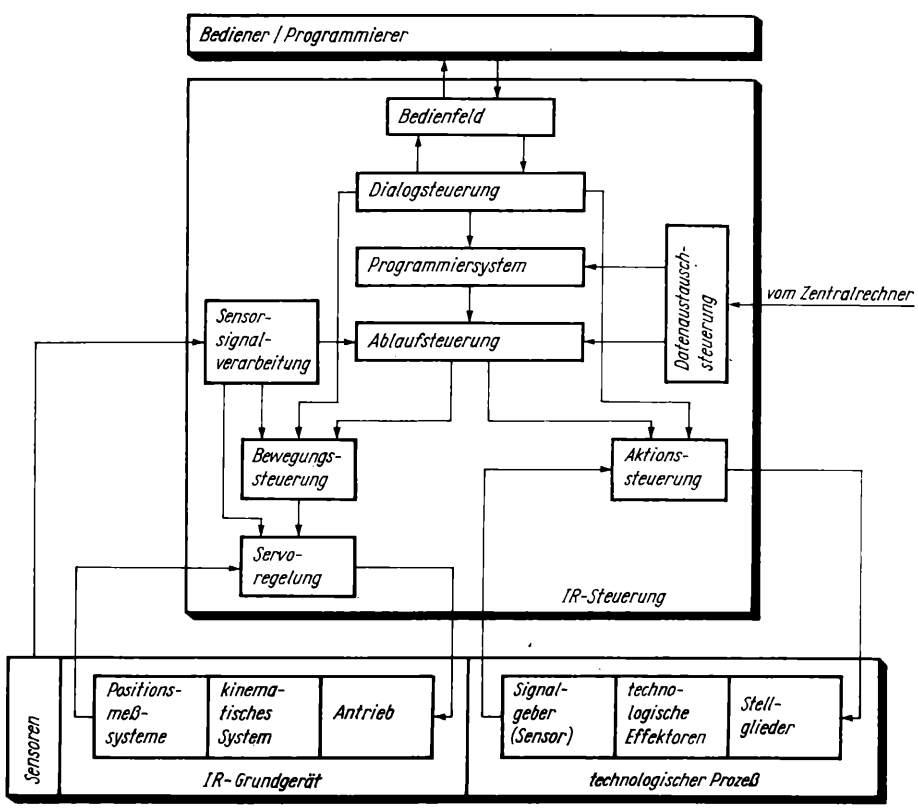


Bild 2.11. Gesamtstruktur einer IR-Steuerung mit wesentlichen Informationsflüssen

archieebenen aus Bild 2.7 und die Zweiteilung der Funktionssteuerung nach Bild 2.8. Mit „IR-Grundgerät" (im Bild unten links) ist der kinematische Teil des Roboters bezeichnet, der die Bewegung ausführt, und unter den „technologischen Effektoren" (im Bild unten rechts) sollen alle Elemente verstanden werden, die technologisch relevante Aktionen ausführen. Solche Elemente können sich am Grundgerät selbst (Greifer, Schweißzange) oder in der Peripherie (automatische Spanneinrichtung, Schwenktisch) befinden.

Die Dialogsteuerung enthält Funktionsgruppen zur

- Kommunikation mit dem Bediener/Programmierer über das Bedienfeld,
- Wahl bzw. Einstellung der Betriebsart und
- Überwachung und Diagnose des Gesamtsystems Grundgerät–Steuerung–Peripherie.

Diese Funktionsgruppen sind bei verschiedenen Steuerungstypen unterschiedlich ausgebaut und können auch ganz fehlen; die beiden ersten Funktionen sind jedoch immer vorhanden.

Die Betriebsarten der Industrierobotersteuerung werden je nach Steuerungstyp- und -hersteller unterschiedlich festgelegt. Als Hauptbetriebsarten finden sich in der Regel

- Programmieren,
- Arbeiten und
- Halt.

Bei anspruchsvolleren Steuerungen gehört zu jeder Hauptbetriebsart ein mehr oder weniger gestaffeltes System von Unterbetriebsarten. Einige Möglichkeiten sollen nachstehend aufgeführt werden.

Programmieren

Diese Hauptbetriebsart dient zur Erzeugung eines ablauffähigen Roboterprogramms im Arbeitsspeicher der Programmsteuerung. Sie kann in die Unterbetriebsarten

- Programmeingabe, bei der das eigentliche Programmieren erfolgt oder das Programm von einem externen Datenträger (z. B. Magnetbandkassette) eingegeben wird;
- Programmtest und -korrektur, bei denen über das Bedienfeld Zugriff zum Arbeitsspeicher besteht und einzelne Befehle geändert, hinzugefügt oder aus dem Programm entfernt werden können. Daneben kann das Programm schrittweise unter Bedienerführung bei wahlweise zu- oder abgeschalteten Antrieben und technologischen Effektoren ausgeführt werden.

Arbeiten

In dieser Hauptbetriebsart wird das Roboterprogramm ausgeführt. Hier werden Unterbetriebsarten wie

- Automatik (das Programm wird automatisch und wiederholt abgearbeitet),
- Einzelschritt (das Programm wird schrittweise abgearbeitet; für jeden neuen Programmschritt ist eine Bedienhandlung erforderlich),
- Handbetrieb (Bewegungen und Aktionen des Roboters können durch Handeingabe entsprechender Anweisungen direkt ausgeführt werden)

unterschieden. Die beiden letztgenannten Betriebsarten überdecken sich teilweise mit denen zum Programmtest.

Halt

In dieser Betriebsart steht der Industrieroboter. Sie kann über Programmbefehl oder vom Bedienfeld angewiesen oder aber durch ein externes Signal (bei Havariegefahr Not–Halt) herbeigeführt werden.

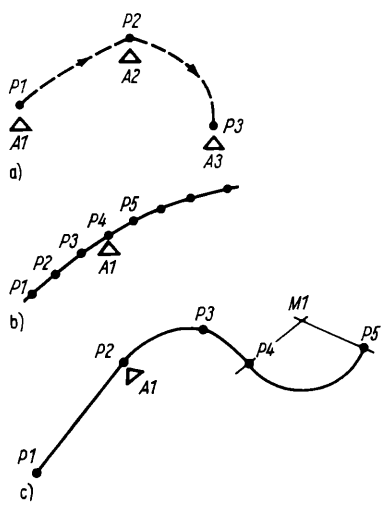
2.5. Einteilung von Industrierobotersteuerungen

Die Vielzahl der verschiedenen Arten von Industrierobotern können nach unterschiedlichen Gesichtspunkten klassifiziert werden [2.5] [2.6]. Eine Möglichkeit, die vom Einsatzfall ausgeht, ist mit ihren Konsequenzen für die einzusetzenden Steuerungen im Abschnitt 2.1. (s. auch Bild 2.2) besprochen worden. Nachfolgend soll eine Möglichkeit der Klassifizierung von IR-Steuerungen behandelt werden, die sich auf den Charakter der Bewegungssteuerung und die Programmierungsart bezieht. Diese Einteilung, die bezüglich der Auslegung einzelner Funktionsgruppen und der Gesamtstruktur Konsequenzen nach sich zieht, hat sich – wenn auch mitunter mit verschiedener Nomenklatur – allgemein eingeführt (z. B. [2.7] [2.8] [2.10] [2.11]).

Danach werden IR-Steuerungen mit

- Punkt-zu-Punkt-Verhalten (PTP-Steuerungen) und
- Bahnverhalten

unterschieden.



1. START BEI P1
2. GEHE NACH P2
3. WENN P2 ERREICHT, AKTION A2
4. WENN A2 BEENDET, GEHE NACH P3
5. WENN P3 ERREICHT, AKTION A3
⋮

1. START BEI P1
2. GEHE NACH P2
3. GEHE NACH P3
4. GEHE NACH P4
5. AKTION A1
6. GEHE NACH P5
⋮

1. START BEI P1
2. GERADE BIS P2
3. AKTION A1
4. PARABEL DURCH P3 BIS P4
5. KREIS U7 M1 BIS P5
⋮

Bild 2.12. Klassifikation von IR-Steuerungen nach dem Bewegungsverhalten
a) Punkt-zu-Punkt-Steuerung (PTP); b) Multipunkt-Steuerung (MP); c) Bahnsteuerung (CP)

Punktsteuerungen gestatten das Abspeichern und sequentielle Anfahren einer Folge diskreter Raumpunkte. Bild 2.12a zeigt dazu eine mögliche Bewegungs- und Aktionsfolge mit Programmbeispiel in umgangssprachlicher Formulierung. Die Bewegungsbahn zwischen den einzelnen Punkten ist nicht definiert; die Achsen fahren dort ohne Funktionszusammenhang. Punktsteuerungen finden vorwiegend bei Handhabegeräten und Punktschweißrobotern Einsatz (s. Abschn. 2.1.).

Die einzelnen Punkte werden auf die Positioniergenauigkeit des Roboters genau angefahren. Erst nach genauer Positionierung auf den programmierten Punkt wird der nächste Befehl aufgerufen. Mitunter ist diese Präzision nicht erforderlich (z.B. beim Umfahren von Hindernissen), und der Zeitverlust für den Positioniervorgang ist eher störend. Für diesen Fall bieten einige PTP-Steuerungen die Möglichkeit der Verwendung von Hilfspunkten (Überschleifen); dabei wird nach Erreichen einer definierten Umgebung des Hilfspunktes die Bewegung nicht unterbrochen, sondern mit der vorgegebenen Geschwindigkeit zum Folgepunkt weitergefahren. Ist im Bild 2.12a z.B. P2 ein solcher Hilfspunkt (A1 möge entfallen), dann erhält Befehl Nr.2 die Form

2. GEHE ÜBER HILFSPUNKT P2.

Eine *Steuerung mit Bahnverhalten* gestattet die Programmierung und das Verfahren definierter Bewegungsbahnen mit Funktionszusammenhang der Bewegungen in den einzelnen Achsen. Dafür bestehen zwei Möglichkeiten:

Multipunkt- (MP-) Steuerungen enthalten das Bewegungsprogramm in Form einer dichten Sequenz von Raumpunkten, die in rascher zeitlicher Folge (etwa alle 10 bis 100 ms) an die Lageregler der Achsen ausgegeben werden. Die Trägheit der Antriebssysteme sorgt für ein ruckfreies Verfahren der Achsen, deren Funktionszusammenhang durch den Synchronismus der Vorgaben der achsspezifischen Sollwerte gewährleistet wird. Bild 2.12b veranschaulicht die Vorgänge. Die Aktion A1 wird dabei ausgelöst, nachdem die Koordinatenwerte von P4 an die Lageregler ausgegeben worden sind, und zwar unabhängig davon, ob diese Position bereits erreicht worden ist oder nicht. Der dadurch auftretende geringe Zeitverzug kann in der Regel in Kauf genommen werden.

Multipunktsteuerungen werden vorwiegend für Farbspritzroboter, aber auch beim Lichtbogenschweißen und bei der Oberflächenbearbeitung (Schleifen, Polieren) eingesetzt. Dem Nachteil des höheren Speicherplatzbedarfs durch das Abspeichern einer Vielzahl von Raumpunkten kann durch geeignete Verfahren zur Informationsverdichtung bei der Aufnahme und Datendekompression bei der Wiedergabe bis zu einem gewissen Grad begegnet werden.

Bahn- (CP-) Steuerungen gestatten das Verfahren mathematisch definierter Bewegungsbahnen. Bild 2.12c zeigt dazu ein Beispiel. Die Programmierung erfolgt im indirekten Teach-in- oder textuell (s. Abschn. 6.2.). Das Roboterprogramm steht in der im Bild 2.12c angedeuteten Form kodiert im Speicher. Ein Bahnkurvenrechner (Interpolator) ermittelt in Auswertung des jeweiligen Bewegungsbefehls eine Anzahl von Zwischenwerten auf der angewiesenen Bahnkurve und gibt sie entsprechend der programmierten Geschwindigkeit an die Lageregler aus. Typische Einsatzfälle für CP-Steuerungen sind Lichtbogenschweißen, Gußputzen und Entgraten.

2.6. Industrierobotersteuerungen auf Mikrorechnerbasis

Bei der Realisierung von elektronischen Steuerungen für Industrieroboter hat sich der Mikrorechner als universelles und flexibles informationsverarbeitendes System durchgesetzt. Die Hardwarestruktur, d.h. die gerätetechnische Konfiguration, ist dabei weitgehend standardisiert. Eine für eine IR-Steuerung mögliche Variante ist im Bild 2.13 angegeben. Sie besteht im Kern aus einer zentralen Verarbeitungseinheit (ZVE), die im wesentlichen aus einem Mikroprozessor gebildet wird, dem Festwertspeicher (ROM), der die Arbeitsprogramme enthält, und dem Schreib-Lese-Speicher (RAM), in den zeitweilig Programmteile und Daten eingetragen werden. Diese Funktionseinheiten sind

untereinander und mit der Bedien- und Programmierperipherie (im Bild oben rechts) und der Prozeßperipherie (unten) durch ein System von Datensammelleitungen (Bus) verbunden. Die eigentliche Funktion einer Robotersteuerung erhält das System durch die Software, d.h. durch die Gesamtheit der Programme, die in den Arbeitsspeicher (ROM und RAM) des Rechnerkerns eingetragen werden. Dazu wird der Funktionsinhalt der IR-Steuerung in Teilfunktionen zerlegt und für jede dieser Funktionen ein Softwaremodul entwickelt. Eine Übersicht über solche Funktionsmodule ist im Bild 2.14 zusammengestellt [2.13]. Diese Funktionsmodule sind nicht voneinander unabhängig, sondern größtenteils miteinander funktionell verbunden. So muß z.B. der Funktionsmodul „Synchronisierung" Einfluß auf den Steuerungsablauf nehmen, und die Sollwerte für die Adreßbewegung müssen den Servoreglern im richtigen Zeitpunkt zur Verfügung gestellt werden. Einige der Verarbeitungsaufgaben, wie gerade die Servoregelung, stellen Echtzeitforderungen und müssen in einem vorgegebenen Zeitraster bearbeitet werden; andere Prozesse, wie die Kollisionsraumüberwachung, müssen nahezu ständig laufen und konkurrieren um die Bearbeitung durch den Prozessor.

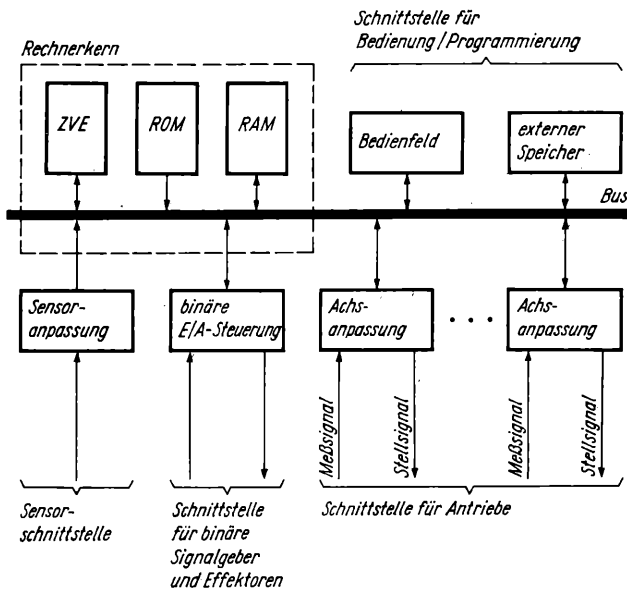


Bild 2.13
Hardwarestruktur einer Steuerung auf Mikrorechnerbasis

Da die ZVE entsprechend dem Rechnergrundprinzip (s. Abschn. 2.4.2.) die Befehle eines Programms nur sequentiell verarbeiten kann, sorgt ein spezielles Programm, das Betriebssystem, für eine koordinierte Bearbeitung der einzelnen Softwaremodule. Häufig reicht die Verarbeitungsleistung eines einzelnen Mikroprozessors zur Realisierung aller Steuerungsfunktionen unter Gewährleistung der Echtzeitforderungen nicht aus. Dann kommen Mehrprozessor- bzw. Mehrrechnerstrukturen zum Einsatz, bei denen die benötigten Softwaremodule geeignet auf die einzelnen verarbeitenden Komponenten verteilt werden müssen.

Sind das Betriebssystem und die erforderlichen Softwaremodule erstellt und im Arbeitsspeicher des Mikrorechners implementiert, so hat das Gesamtsystem die gewünschte Funktion einer Robotersteuerung. Das stellt sich sehr anschaulich im sog. Schalenmodell (Bild 2.15) dar. Der Rechnerkern wird durch das Betriebssystem und durch die Softwaremodule umhüllt, und dem Anwender ist nur die äußere Hülle zugänglich. Er bedient und programmiert also nicht den Mikrorechner (der für ihn in der Regel gar nicht zugänglich

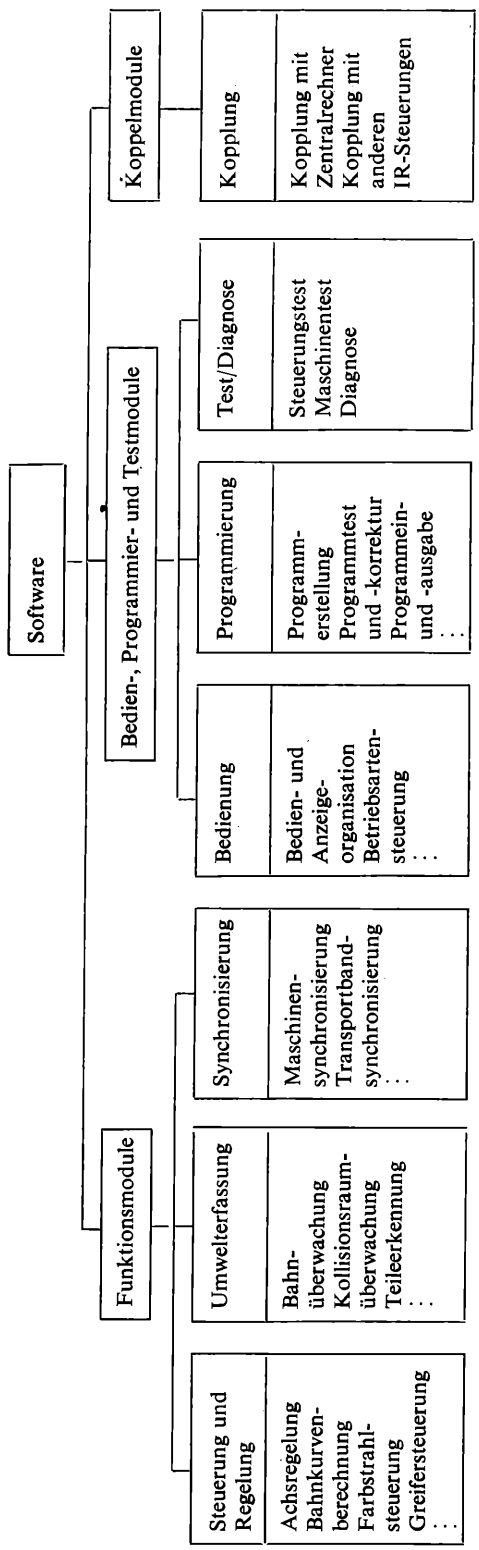


Bild 2.14. Softwaremodule einer IR-Steuerung auf Mikrorechnerbasis (nach [2.13])

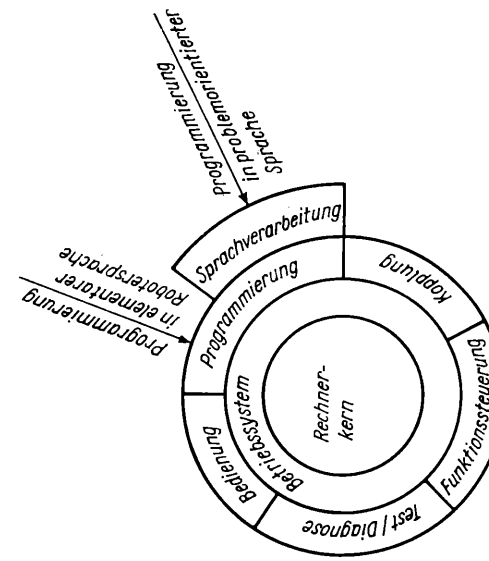


Bild 2.15
Schalenmodell einer Mikrorechnersteuerung

ist), sondern eine programmtechnisch realisierte IR-Steuerung. Wird eine Rechnerstruktur wie im Bild 2.7 über Software auf einer anderen Rechnerstruktur (hier der Mikrorechner) realisiert, so spricht man auch von einer virtuellen Maschine. Dabei sind mindestens zwei Niveaus der Programmierung und damit auch der Programmiersprachen zu unterscheiden. Hier sind es

- für den Mikrorechner die mikrorechnertypische Programmiersprache und
- für die Robotersteuerung (virtuelle Maschine) die Gesamtheit der robotertypischen Befehle einschließlich der Regeln ihrer Aufeinanderfolge (Grammatik).

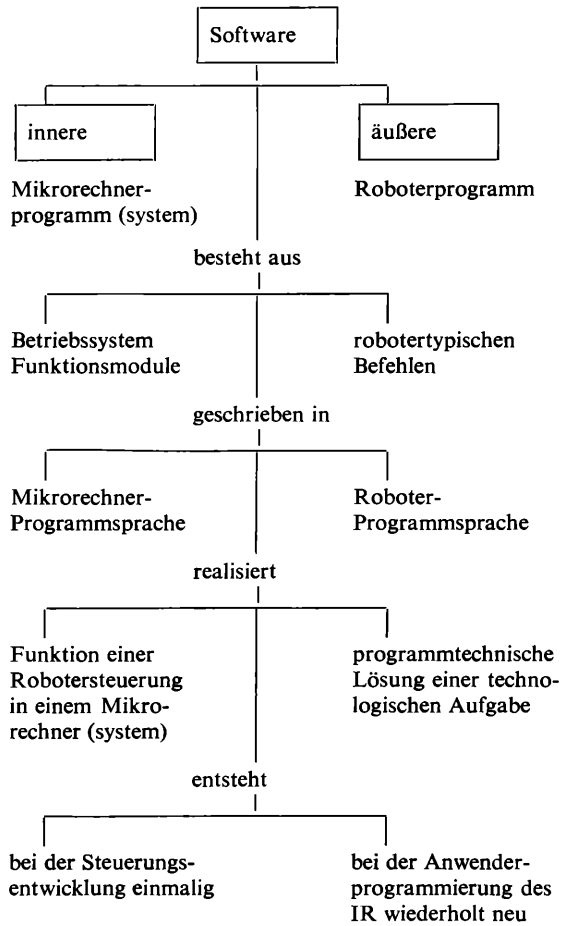


Bild 2.16
Innere und äußere Software für eine IR-Steuerung auf Mikrorechnerbasis

Eine weitere virtuelle Maschine und entsprechend eine weitere Sprachebene entsteht, wenn im Programmiersystem ein Sprachverarbeitungsmodul zur Verarbeitung einer problemorientierten Robotersprache vorgesehen ist. Er ist, wie im Bild 2.15 angedeutet, Bestandteil einer weiteren Hülle im Schalenmodell.

Die beiden grundsätzlichen Sprachebenen einer Industrierobotersteuerung mit einigen ihrer Merkmale sind abschließend im Bild 2.16 gegenübergestellt.

3. Kinematische Systeme, Antriebe und Meßsysteme

3.1. Allgemeines

Die Bewegung eines Roboters wird durch die Antriebe bewirkt, vom kinematischen System ausgeführt und mit den Meßsystemen erfaßt. Für die Auslegung der Bewegungssteuerung ist die Kenntnis der Wirkprinzipien und des Verhaltens dieser Funktionseinheiten erforderlich. Die Grundstruktur ihrer Verkopplung ist im Bild 3.1 dargestellt. In Abhängigkeit von den Ansteuersignalen wirken einerseits die Antriebe mit Kräften und Momenten auf das kinematische System ein, das darauf mit Bewegungen reagiert; außerdem wirken noch andere Kräfte (Schwerkraft, Reibung) auf das kinematische System ein. Andererseits wirkt das bewegte kinematische System auf die Antriebe zurück.

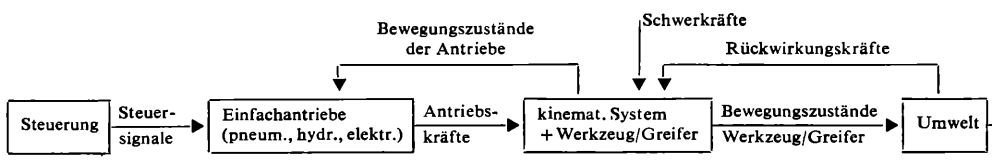


Bild 3.1. Antriebe, kinematisches System und Meßsysteme eines Industrieroboters und ihre Verkettung

Ziel dieses Abschnitts ist es, Grundlagen für die mathematische Beschreibung der Teile des Industrieroboters zu erarbeiten, auf deren Verhalten die optimale Auslegung der Bewegungssteuerung ausgerichtet sein muß. Die mathematische Beschreibung der obengenannten Funktionseinheiten wird unter dem Aspekt der Modellbildung des dynamischen Verhaltens der Teilsysteme Kinematik, Antriebe, Meßglieder und ihrer Wechselwirkung vorgenommen. Die erforderliche Güte des Modells und der damit verbundene Aufwand bei der Modellierung hängen wesentlich von der durch die technologische Aufgabe bedingten Präzision der Bewegungssteuerung ab. Quantitative und qualitative Aussagen über die Güte des Modells erhält man durch Vergleich von Simulationsergebnissen mit denen praktischer Messungen.

Die Modelle der einzelnen Funktionseinheiten müssen dazu aufeinander abgestimmt und aneinander anschließbar sein. Einfach ist in diesem Zusammenhang das Anpassen der Meßglieder, da deren Eingangs- und Ausgangsgrößen festliegen und i. allg. Rückwirkungsfreiheit angenommen werden kann. Größere Komplexität weisen die Teilsysteme Kinematik und Antriebe sowohl in ihrer inneren Struktur als auch in ihrer Wechselwirkung auf.

Für die folgenden Betrachtungen wird dazu vorausgesetzt:

1. Die Kinematikglieder sind federnd miteinander verbunden, während Kräfte direkt ins Gestell ohne Nachgiebigkeit eingeleitet werden.
2. Die Antriebe wirken direkt auf diese Glieder ein. Die bewegte Antriebsmasse wird den Kinematikgliedern zugeschlagen, während Reibkräfte auf die Antriebe konzentriert

werden, da zum einen die Trägheit des Kinematiksystems meist überwiegt und zum anderen im Antrieb eine Abhängigkeit der Antriebskraft von seiner Geschwindigkeit ohnehin gegeben ist.

3. Die Beschreibung der Antriebe (im Abschn. 3.3.) ohne Berücksichtigung angelenkter Teile des Kinematiksystems würde zu Fehleinschätzungen des dynamischen Verhaltens führen. Aus diesem Grund werden im Abschnitt 3.2. einfache Glieder des kinematischen Systems so beschrieben, daß ihre Trägheit und die auf sie wirkenden Kräfte auf die Antriebsachse bezogen werden können.

3.2. Funktionseinheiten des kinematischen Systems

3.2.1. Beispiele kinematischer Strukturen

Das vom Industrieroboter gehandhabte Werkstück muß im Raum positioniert und orientiert werden. Die Positionierung geschieht durch die drei Grund- oder Hauptachsen des Industrieroboters, die als Dreh- (D) bzw. Schwenkglieder oder als Schub- bzw. Linearglieder (L) ausgelegt sein können. Die Kombination aller Varianten ergibt $2^3 = 8$ verschiedene Möglichkeiten der kinematischen Strukturen der Grundachsen (Bild 3.2). Die häufigsten Strukturen sind die DDD-Struktur, die DDL-Struktur und auch die in Werkzeugmaschinen überwiegend zu findende LLL-Struktur.

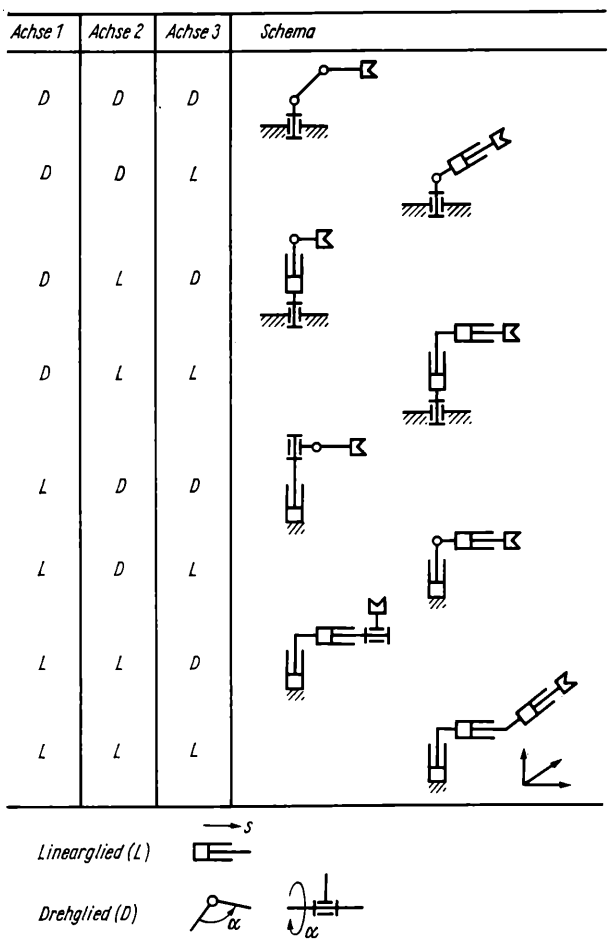


Bild 3.2
Acht Möglichkeiten der kinematischen Struktur der Industrieroboter-grundachsen

Ausgehend von den allgemeinen Bewegungsgleichungen eines starren Kinematikglieds wird im Abschnitt 3.2.3. die Bewegung von Schubgliedern und im Abschnitt 3.2.4. die Bewegung von Schwenkgliedern beschrieben.

3.2.2. Allgemeine Bewegung eines starren Kinematikglieds

Die allgemeine Bewegung eines starren Körpers läßt sich als Überlagerung einer Translation und einer Rotation um den Schwerpunkt darstellen. Folglich müssen zu ihrer Beschreibung zwei Gruppen von Bewegungsgleichungen angesetzt werden.

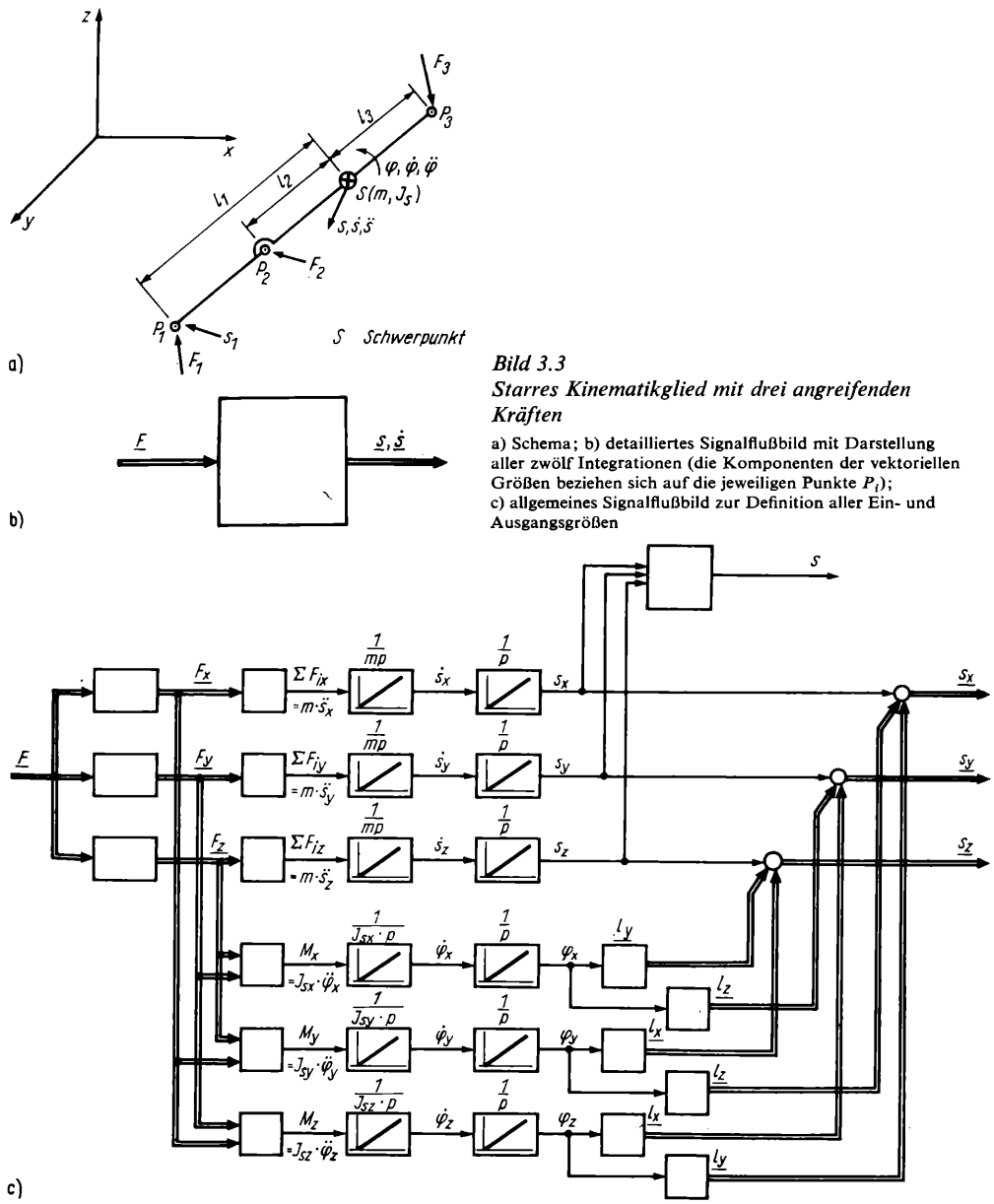


Bild 3.3
Starres Kinematikglied mit drei angreifenden Kräften

a) Schema; b) detailliertes Signalflußbild mit Darstellung aller zwölf Integrationen (die Komponenten der vektoriellen Größen beziehen sich auf die jeweiligen Punkte P_i);
c) allgemeines Signalflußbild zur Definition aller Ein- und Ausgangsgrößen

Generell gelten für ein gerades, starres Kinematikglied mit n Kraftangriffspunkten (Bild 3.3, $n = 3$) ohne Berücksichtigung der Reibkräfte folgende Kräfte- und Momentengleichgewichte:

$$\begin{aligned} m\ddot{s}_x &= \sum F_{ix} \\ m\ddot{s}_y &= \sum F_{iy} \\ m\ddot{s}_z &= \sum F_{iz} \\ J_{Sx}\ddot{\varphi}_x &= \sum L_{iy}F_{iz} + \sum L_{iz}F_{iy} \\ J_{Sy}\ddot{\varphi}_y &= \sum L_{ix}F_{iz} + \sum L_{iz}F_{ix} \\ J_{Sz}\ddot{\varphi}_z &= \sum L_{ix}F_{iy} + \sum L_{iy}F_{ix}; \end{aligned} \tag{3.1}$$

m Masse des Kinematikglieds, im Schwerpunkt konzentriert
J_S Massenträgheitsmoment des Gliedes um den Schwerpunkt S
x, y, z Komponenten in x, y, z-Richtung oder um zu x, y, z parallele Achsen
i lfd. Nummer der Kraftangriffspunkte
s Translation des Schwerpunktes
φ Rotation um den Schwerpunkt
$\ddot{s}$ geradlinige Beschleunigung
$\ddot{\varphi}$ Winkelbeschleunigung.

Die Beziehungen (3.1) gelten auch, wenn ein Moment am Kinematikglied angreift. Es kann dann in ein Kräftepaar zerlegt werden. Dieser Ansatz wird im Abschn. 4.2. angewendet.

Die zweimalige Integration aller sechs Beschleunigungsgrößen im Gleichungssystem (3.1) ergibt die entsprechenden Weg- und Winkeländerungen bezüglich des Schwerpunktes. Bei linearisierter Betrachtung kleiner Schritte sind die Wegänderungen für die Punkte P_i des Kinematikglieds:

$$\begin{aligned} \Delta s_{ix} &= \Delta s_x + L_{iy}\,\Delta\varphi_z + L_{iz}\,\Delta\varphi_y \\ \Delta s_{iy} &= \Delta s_y + L_{ix}\,\Delta\varphi_z + L_{iz}\,\Delta\varphi_x \\ \Delta s_{iz} &= \Delta s_z + L_{ix}\,\Delta\varphi_y + L_{iy}\,\Delta\varphi_x. \end{aligned} \tag{3.2}$$

Günstig darstellbar sind derartige Zusammenhänge in Signalflußbildern, vor allem, wenn zu Verzögerungen führende Integrationen oder Verzögerungsglieder enthalten sind (Bild 3.3 b), da Grundzusammenhänge (hier für Beschleunigungen $\ddot{s}$, $\ddot{\varphi}$, Geschwindigkeiten $\dot{s}$, $\dot{\varphi}$, Wege s bzw. Winkel φ) und Bilanzen (hier für Wege, später für Kräfte, Ströme usw.) sowie die Wirkungswege gut erkennbar sind. Blöcke mit integrierender Wirkung sind an der Integration einer Sprungfunktion im jeweiligen Block und am reziproken Differentialoperator $1/p$ erkennbar.

Im Bild 3.3 c ist das allgemeine Signalflußbild zur Definition der Ein- und Ausgangsgrößen angegeben.

3.2.3. Schubglied

Ein Schubglied beschreibt eine allgemeine Bewegung, wenn sich zwischen ihm und dem Gestell weitere bewegte Kinematikglieder befinden. Einfach zu beschreiben ist der Fall, daß sich das Schubglied auf einer in Ruhe befindlichen, steifen Unterlage befindet (Bild 3.4).

Die einfache Darstellung resultiert vor allem daraus, daß sich Trägheit und Schwerkraft nicht mit dem Weg s verändern.

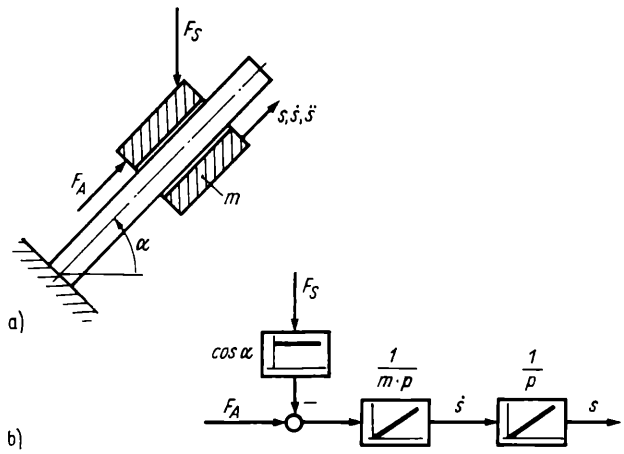


Bild 3.4
Schubglied
a) Schema; b) Signalfluß (Reibung dem Antrieb zugeschlagen) F_A antreibende Kraft; F_S Schwerkraft

3.2.4. Schwenkglied

Bei der Beschreibung eines solchen Gliedes ergeben sich trotz Annahme des Spezialfalls, daß sich das übrige kinematische System in Ruhe befindet und ideal steif ist, kompliziertere Zusammenhänge als beim Schubglied. Im Bild 3.5 sind drei Strukturen eines Schwenkglieds angegeben:

Struktur S1: Angriffspunkt einer antreibenden Kraft F_A zwischen Schwerpunkt P und Schwerpunkt S

Struktur S2: Angriffspunkt von F_A außerhalb der Strecke $\overline{PS}$

Struktur S3: Angriffspunkt eines Schwenkmoments M_A im Punkt P.

Die Momentengleichgewichte sind wie folgt anzugeben:

$$\text{S1 und S2: } (J_S + ml_S^2)\,\ddot{\varphi}_1 = F_A r_A - F_S r_S - F_L r_L \tag{3.3}$$

$$\text{S3: } (J_S + ml_S^2)\,\ddot{\varphi}_1 = M_A - F_S r_S - F_L r_L \tag{3.4}$$

(Nach dem Satz von *Steiner* gilt $J_P = J_S + ml_S^2$.)
In diesen Gleichungen sind

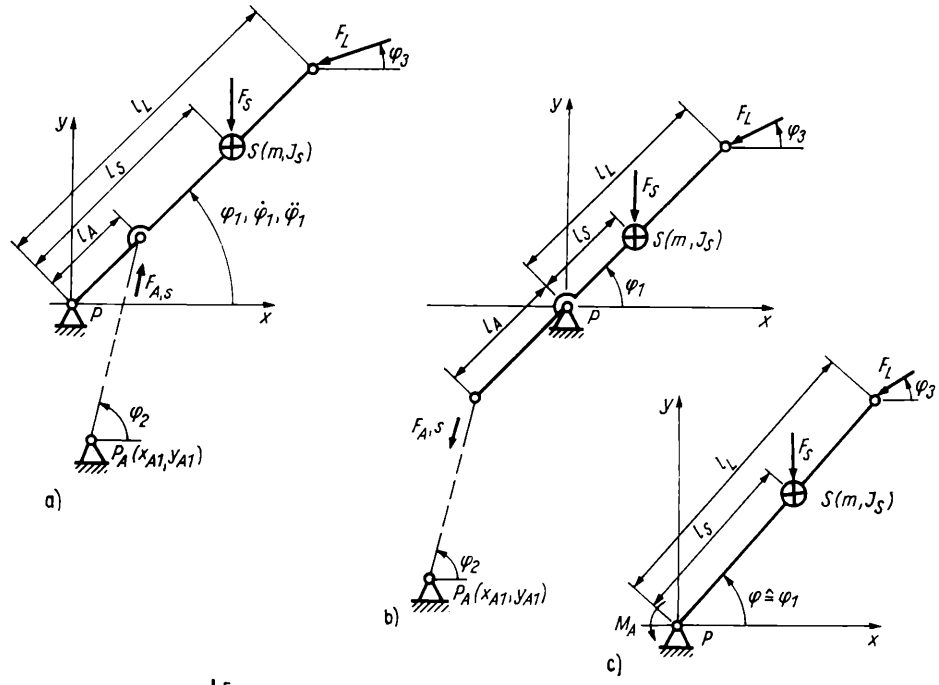
r_A Kraftarm
r_S, r_L Lastarme
l_S Trägheitsarm
J_P, J_S Massenträgheitsmomente bezüglich Punkt P und Schwerpunkt S.

Die Kraft- und Lastarme ändern sich – das ist aus Bild 3.5a bis c zu erkennen – in Abhängigkeit von der Lage des Schwenkglieds, während der Trägheitsarm l_S bei konstantem Schwerpunkt konstant bleibt.

Für die Lastarme sind folgende einfache Beziehungen für S1, S2, S3 anzugeben:

$$\begin{aligned} r_S &= l_S \cos\varphi_1 \\ r_L &= l_L \cos(90° + \varphi_1 - \varphi_3) = l_L \sin(\varphi_3 - \varphi_1). \end{aligned} \tag{3.5}$$

Es sei darauf hingewiesen, daß die Lastarme negative Werte annehmen können (z. B. bei $\varphi_1 > \varphi_3$).



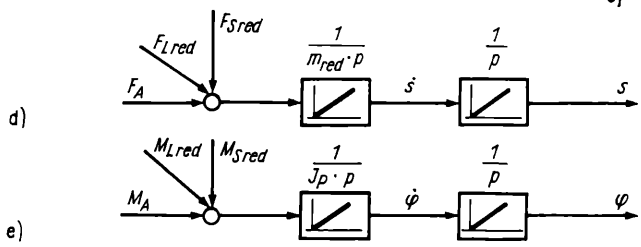


Bild 3.5
Schwenkglied

a), b), c) verschiedene Strukturen; d), e) Signalflußbilder der auf die Antriebsachsen reduzierten Größen (im Arbeitspunkt linearisiert)

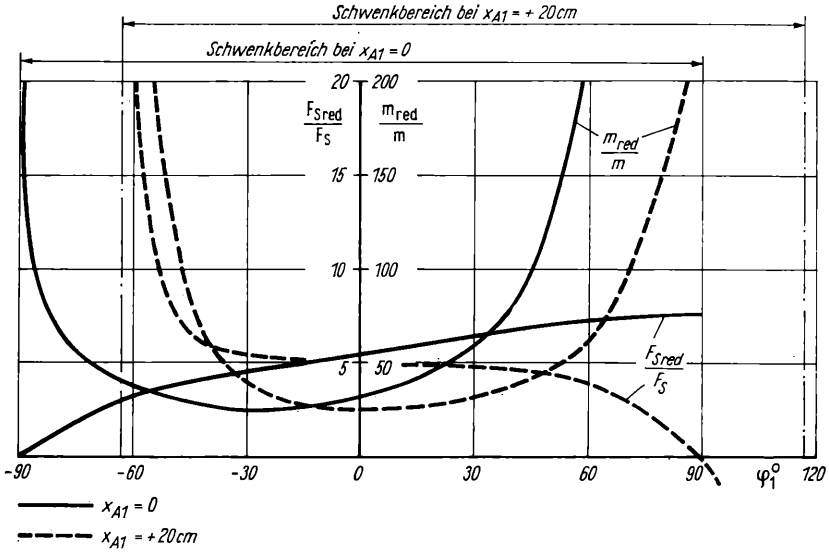


Bild 3.6. Abhängigkeit reduzierter Massen und Kräfte eines Schwenkgliedes vom Arbeitspunkt

$F_{S\,red}$ auf Antriebsachse reduzierte Schwerkraft; m_{red} reduzierte bewegte Masse

Für den Kraftarm der Strukturen S1 und – in Klammern – S2 gilt

$$r_A = L_A \sin(\varphi_2 - \varphi_1) \tag{3.6}$$

$$\varphi_2 = \arccos \frac{\pm l_A \cos\varphi_1 - x_{A1}}{\sqrt{(\pm l_A \cos\varphi_1 - x_{A1})^2 + (\pm l_A \sin\varphi_1 - y_{A1})^2}}.$$

Im Bild 3.5a sind $x_{A1} > 0$, $y_{A1} < 0$ und im Bild 3.5b $x_{A1} < 0$, $y_{A1} < 0$.

Werden diese Zusammenhänge auf die Achse der antreibenden Kraft bzw. des antreibenden Moments bezogen – beachte $d_s/d\varphi_1 = r_A$! –, so ergeben sich folgende reduzierte Größen:

$$\text{S1, S2: } m_{red} = \frac{J_S + m l_S^2}{r_A^2}$$

$$F_{S\,red} = F_S \frac{r_s}{r_A} \tag{3.7}$$

$$F_{L\,red} = F_L \frac{r_l}{r_A}$$

$$\text{S3: } J_P = J_S + m l_S^2$$

$$M_{S\,red} = F_S r_S \tag{3.8}$$

$$M_{L\,red} = F_L r_L.$$

Für die Struktur S1 wurden zwei Beispielfälle berechnet mit jeweils $l_A = 20$ cm' $l_S = 100$ cm, $y_{A1} = 40$ cm sowie $x_{A1} = 0$ cm und $x_{A1} = +20$ cm (J_S wurde vernachlässigt, F_L wurde nicht berücksichtigt). Die Ergebnisse sind im Bild 3.6 dargestellt.

Es ist zu erkennen, daß in einem großen Bereich des theoretisch möglichen $\Delta\varphi_1$ von 180° die reduzierten Massen mehr als zwei Größenordnungen über den tatsächlich bewegten liegen und daß der Verlauf der reduzierten Schwerkraft für $x_{A1} = 0$ günstiger ist als für $x_{A1} = +20$ cm. Bei der Konstruktion der Kinematik und bei der statischen und dynamischen Auslegung der Antriebe spielen derartige Betrachtungen eine wichtige Rolle.

3.3. Antriebe für das kinematische System

In den Industrierobotern kommen elektrische, hydraulische und pneumatische Antriebe zum Einsatz. Für einfache oder sich selten ändernde Handhabungsaufgaben können sie sehr einfach aufgebaut sein. Die freie Programmierbarkeit bei hoher Auflösung des Verfahrbereichs setzt jedoch, unabhängig von der gewählten Hilfsenergie, meist eine Lageregelung dieser Antriebe voraus. Bei Beachtung der Grenzen des dynamischen Verhaltens ist auch der Einsatz von Schrittantrieben möglich. In Tafel 3.1 sind, ausgehend von den Handhabungsaufgaben, verschiedene Antriebsarten gegenübergestellt.

Ausführlich wird auf die in Lageregelungen eingesetzten elektrohydraulischen und elektrischen Antriebe eingegangen. Die Erarbeitung von Modellen ihres dynamischen Verhaltens ist vor allem als Grundlage der im Abschnitt 4. behandelten Bewegungssteuerungen notwendig, da diese Antriebe für die Realisierung auch dynamisch anspruchsvoller Handhabeaufgaben am geeignetsten sind.

Tafel 3.1. Eignung typischer Antriebsvarianten für bestimmte Handhabungsaufgaben

Handhabungsaufgabe	Lagegeregelter Antrieb	Schrittantrieb	Einfachantrieb
Handgesteuerter Synchronmanipulator	geeignet	nicht geeignet	nicht geeignet
Festprogrammierter Manipulator	zu aufwendig	geeignet, aber meist zu aufwendig	gut geeignet, pneumatischer Einfachantrieb meist am wirtschaftlichsten
Textuell programmierbarer Industrieroboter	vor allem bei hoher Auflösung des Verfahrbereiches wirtschaftlich	gut geeignet	nur geeignet, wenn viele Anschläge oder Endschalter, deren Wirksamsein programmiert werden kann, im Verfahrbereich vorhanden sind
Industrieroboter, indirekt teach-in-programmierbar (s. Abschn. 6.)	geeignet	bedingt geeignet	nicht einsetzbar
Industrieroboter, direkt teach-in-programmierbar bei kraftlosen Antrieben	geeignet: elektrohydraulische Antriebe bedingt geeignet: pneumatische mit elektrischer Bremse nicht geeignet: elektrische und elektro-pneumatische wegen sehr hoher Getriebeübersetzungen	nicht einsetzbar	nicht einsetzbar
Industrieroboter, direkt trach-in-programmierbar mit Hilfe der Antriebe	geeignet	bedingt geeignet	nicht einsetzbar

3.3.1. Elektrohydraulische Antriebe für Lageregelungen

3.3.1.1. Struktur der Antriebe und der Druckquelle

Im Bild 3.7 ist schematisch die Struktur des Teiles Druckquelle, Steuerventile, Verbraucher eines Industrieroboters dargestellt. Zur Realisierung von Lageregelungen sind Stetigsteuerventile einzusetzen. In allen modernen Hydraulikkonzeptionen versorgt eine zentrale Druckquelle alle Verbraucher einer Anlage.

Der Name Druckquelle steht dabei (analog zur Spannungsquelle) für die Sicherung annähernder Druckkonstanz auch bei sich ändernder Belastung durch die Verbraucher. Die Druckkonstanz wird bei hohen Ansprüchen an zulässige Druckschwankungen nach geringen Eigenverlusten der Druckquelle mit Hilfe einer Verstellpumpe mit Druckregelung realisiert. Als Pumpe werden Flügelzellen-, Radial- und Axialkolbenpumpen eingesetzt. Als Akkumulator zur Abdeckung von Belastungsspitzen und zur günstigen Gestaltung des dynamischen Verhaltens der Druckquelle (s. Abschn. 3.3.1.4.) werden Druckflüssigkeitsspeicher (Gasblasenspeicher, Kolbenspeicher, Federspeicher) eingesetzt. Selten er-

füllen Konstantpumpenkombinationen die Anforderungen an die Regelgüte für den Druck in Industrierobotern.

Im Bild 3.7 ist angedeutet (s. gestrichelte Leitungsführungen), daß

- die Druckregelung direkt oder mit Vorsteuerung erfolgen kann und
- eine ausreichend große Druckquelle die Antriebe mehrerer Anlagen, also ggf. auch mehrerer Industrieroboter, versorgen kann, wie viele Beispiele bereits zeigen.

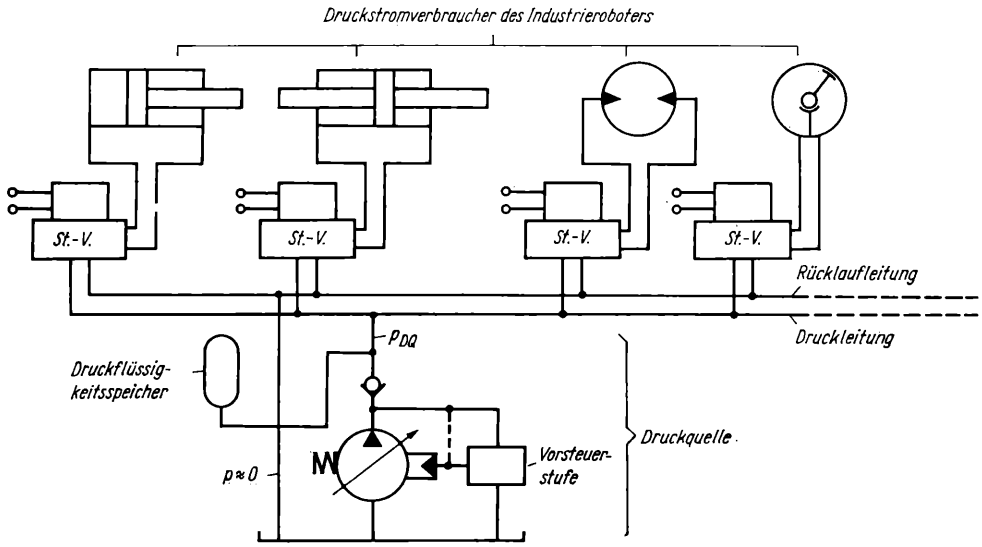


Bild 3.7. Struktur des Teilsystems Druckwelle, Steuerventile, Druckstromverbraucher eines hydraulisch angetriebenen Industrieroboters (dargestellt sind vier verschiedene Druckstromverbraucher)

Die Antriebsschaltungen aus Steuerventil und Druckstromverbraucher liegen parallel am Druckquellendruck p_{DQ}. Zusätzliche mechanische Getriebe sind für derartige Antriebe nicht typisch.

3.3.1.2. Bestandteile der Antriebe und ihre Modelle

Die wesentlichen Grundbeziehungen für die Modellierung des dynamischen Verhaltens hydraulischer Baugruppen (s. auch [3.2] und vgl. die folgenden Bezeichnungen mit denen im Bild 3.8a) sind

- die an den Flächen A_i wirkenden Druckkräfte F_i infolge der Drücke p_i:

$$F_i = A_i p_i \tag{3.9a}$$

- die bei Kolbengeschwindigkeit s entstehenden Verschiebeölströme Q_{Vei}:

$$Q_{Vei} = A_i s \tag{3.9b}$$

- die Proportionalität zwischen gespeichertem Ölstrom Q_{gi} und $p_i = \mathrm{d}p/\mathrm{d}t$, der Änderungsgeschwindigkeit des Druckes p_i im Ölvolumen V_i:
  Der Proportionalitätsfaktor ist die hydraulische Kapazität C_y

$$C_y = \frac{Q_{gi}}{p_i} = V_i \beta . \tag{3.9c}$$

Der Kompressibilitätsfaktor β von Hydrauliköl kann unter normalen Einsatzbedingungen des Industrieroboterantriebs mit $\beta \approx 8 \cdot 10^{-4}\ \mathrm{MPa}^{-1}$ angesetzt werden.

- die Ölstrom-Druckabfall-Beziehungen $Q = f(\Delta p)$:
  in blendenförmigen und ähnlichen Widerständen mit turbulenter Strömung, Strömungsquerschnitt A_{Dr}

$$Q = K_1 A_{\mathrm{Dr}} \sqrt{\Delta p} \tag{3.9d}$$

  in Rohren und Schläuchen des Querschnitts A_{R} und der Länge l, laminare Strömung

$$Q = K_2 \frac{A_{\mathrm{R}}^2}{l} \Delta p \tag{3.9e}$$

  in Leckspalten der Höhe h, der Breite b und der Länge l, laminare Strömung

$$Q = K_3 \frac{bh^3}{l} \Delta p. \tag{3.9f}$$

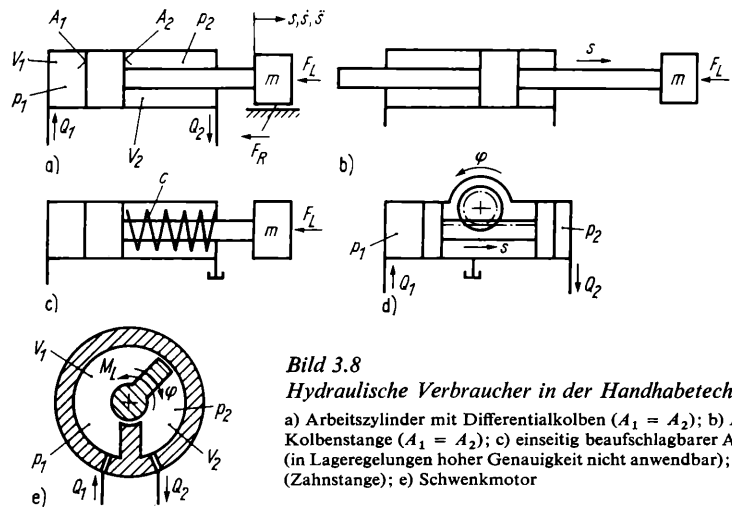


Bild 3.8
Hydraulische Verbraucher in der Handhabetechnik
a) Arbeitszylinder mit Differentialkolben ($A_1 = A_2$); b) Arbeitszylinder mit beiderseitiger Kolbenstange ($A_1 = A_2$); c) einseitig beaufschlagbarer Arbeitszylinder mit Rückstellfeder (in Lageregelungen hoher Genauigkeit nicht anwendbar); d) Arbeitszylinder mit Ritzel (Zahnstange); e) Schwenkmotor

Während K_1 als lediglich von der Dichte des Öles abhängig angesetzt werden kann, sind K_2 und K_3 zusätzlich viskositätsabhängig.

- die an einer Masse m wirkende Kraft F_{B} bei Beschleunigung $\ddot{s}$:

$$F_{\mathrm{B}} = m\ddot{s} \tag{3.9g}$$

- die von der Geschwindigkeit s abhängige Reibkraft F_{R}, im einfachsten Fall geschwindigkeitsproportional mit dem Proportionalitätsfaktor k:

$$F_{\mathrm{R}} = k\dot{s}. \tag{3.9h}$$

Als Bilanzen sind Kräftebilanzen an bewegten Teilen und Ölstrombilanzen in Volumina mit veränderlichem Druck aufzustellen. Während die Grundbeziehungen nur einmal ermittelt werden müssen, hängt die jeweilige Bilanz von den Gegebenheiten jeder Baugruppe ab.

Verbraucher

Im Bild 3.8 sind typische hydraulische Verbraucher für die Handhabetechnik dargestellt. Sie haben in drei Fällen translatorische, in zwei Fällen rotatorische Abtriebsbewegungen.

In jedem Fall sind sie mit dem kinematischen System gekoppelt, so daß Masse m (bzw. Massenträgheitsmoment J) und äußere Belastung F_L (bzw. M_L) i. allg. reduzierte Größen darstellen.

In den Signalflußbildern (Bild 3.9) ist der eigentlich zum kinematischen System gehörende Teil gekennzeichnet. Als Kräftebilanz gilt jeweils

$$m\ddot{s} = A_1 p_1 - A_2 p_2 - F_L - F_R .$$

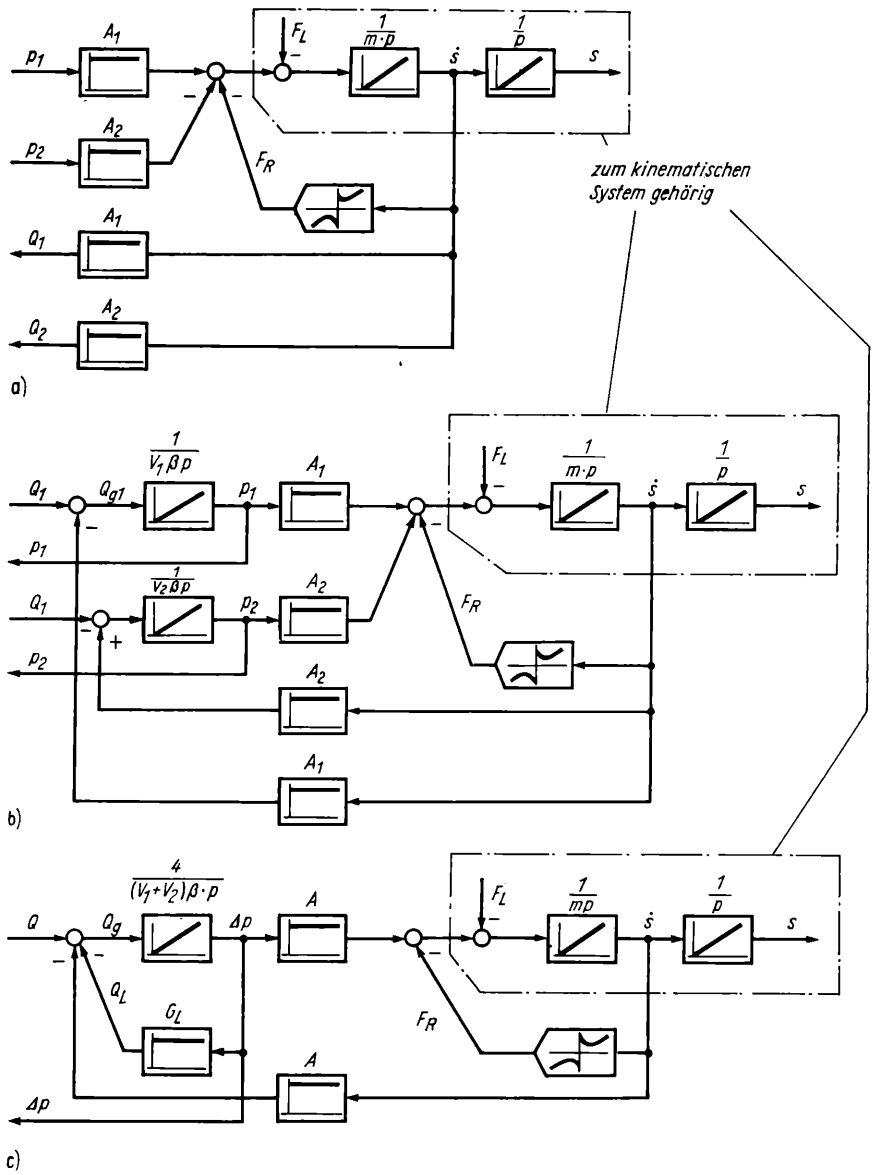


Bild 3.9. Signalflußbilder für Druckstromverbraucher

a) ungleiche Kolbenflächen, Ölkompressibilität und inneres Lecköl vernachlässigbar; b) ungleiche Kolbenflächen, Ölkompressibilität berücksichtigt, inneres Lecköl vernachlässigbar; c) gleiche Kolbenflächen $A_1 = A_2 = A$, $Q_1 = Q_2 = Q$, Ölkompressibilität und innerer Lecköl Q_l berücksichtigt

Wegen $A_1 = A_2 = A$ im Bild 3.9c vereinfacht sich mit $p_1 - p_2 = \Delta p$ das Kräftegleichgewicht dort etwas.

Bild 3.9a zeigt das Signalflußbild eines translatorischen Verbrauchers, bei dem die Kompressibilität der Ölvolumina im Zylinder vernachlässigt wurde, so daß Ölstrombilanzen nicht ermittelt werden mußten. Eingangsgrößen sind die Drücke, Ausgangsgrößen die Ölströme.

Ist diese Vernachlässigung nicht erlaubt, gilt für die Volumina V_1 und V_2 (s. Bild 3.8a):

$$Q_{g1} = Q_1 - A_1\dot{s}$$

$$Q_{g2} = A_2 s - Q_2 .$$

Der Signalfluß des Arbeitszylinders für diesen Fall ist im Bild 3.9b dargestellt. Um Differenzierglieder, die die Simulation der Modelle auf einem Rechner erschweren, zu vermeiden [s. Kapazitätsgrundbeziehung (3.9c)], wurden Eingangs- und Ausgangsgrößen vertauscht.

Das Leitungsvolumen ist dem Verbrauchervolumen zuzuschlagen.

Nicht berücksichtigt wurde in beiden Signalflußbildern die Abhängigkeit des Volumens vom Weg. Die Linearisierung im Arbeitspunkt ist eine i. allg. ausreichende Näherung.

Ebenso ist es nicht notwendig, für die Verbraucher mit rotatorischem Abtrieb spezielle Signalflußbilder bzw. Gleichungssysteme anzugeben. Da meist kleine Volumina vorhanden sind, gilt Bild 3.8a: es müssen lediglich ersetzt werden:

- Fläche A_i durch geometrisches Verdrängungsvolumen V_g (es ist berechenbar im einfachen Fall, aber auch meßbar als Verhältnis von $Q/\dot{\varphi}$ bzw. von $\Delta V/\Delta\varphi$)
- Masse m durch Massenträgheitsmoment J
- alle F_i durch M_i.

Der Arbeitszylinder mit ungleichen Kolbenflächen und großen Volumina besitzt ein relativ kompliziertes Modell (Bild 3.8b). Bei gleich großen Flächen (beiderseitige Kolbenstange) ist das Signalflußbild für den Fall $Q_1 = Q_2 = Q$ auch bei ungleichen Volumina V_1, V_2 vereinfachbar (Bild 3.9c). Da sich im Arbeitszylinder V_1 und V_2 gegensinnig verändern sowie wegen $A_1 = A_2 = A$ die Summe $V_1 + V_2$ konstant bleibt, kann annähernd gesetzt werden:

$$\frac{1}{V_1} + \frac{1}{V_2} \approx \frac{4}{V_1 + V_2} \approx \text{konst.}$$

Im Bild 3.9c wurde der Leckölstrom Q_L von Volumen V_1 nach Volumen V_2 berücksichtigt.

Steuerventil

Im Bild 3.10 ist ein zweistufiges elektrohydraulisches Servoventil dargestellt.

Das elektrische Eingangssignal wird im Magnetsystem in eine Auslenkung der Prallplatte zwischen den zwei Düsen des Doppeldüse-Prallplatte-Systems umgewandelt. Die Düsen bilden mit den zwei Vordrosselventilen eine Brückenschaltung (erste Stufe), in deren Diagonale als Verbraucher der Hauptsteuerschieber liegt. Seine Steuerkanten bilden mit der Hülse vier Steuerspalte, die ebenfalls als Brücke wirken (zweite Stufe).

In ihrer Diagonale liegt der Verbraucher.

Die Zuordnung der Lage des Hauptsteuerschiebers zum Eingangssignal wird in der dargestellten Ventilvariante durch die Rückführfeder realisiert. Andere international angewendete Varianten sind die elektrische Lagemessung, Federn zwischen Kolben und

Gehäuse oder mit der Auslenkung des Steuerschiebers sich ändernde Widerstände in der Vorsteuerbrücke.

Mit Hilfe eines elektrischen Eingangssignals von weniger als 1W können bis zu 10 kW Ausgangsleistung gesteuert werden.

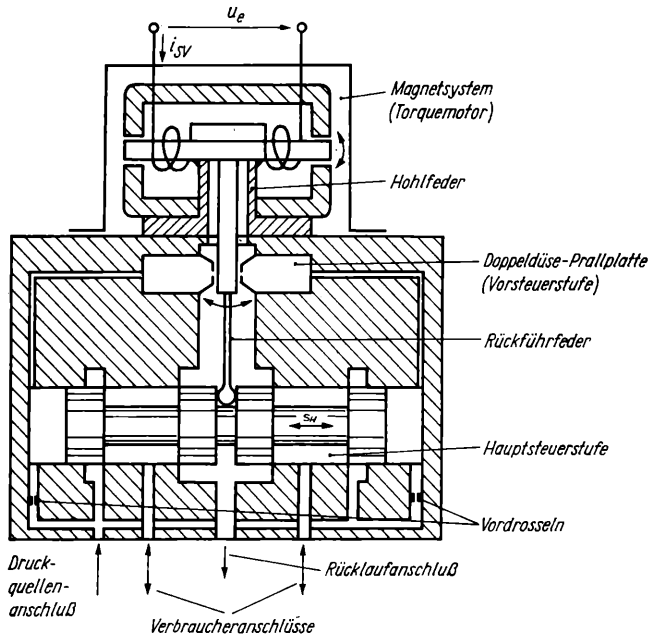


Bild 3.10
Zweistufiges elektrohydraulisches Servoventil (prinzipieller Aufbau) [3.2]

Meist geringere Verstärkungen bei auch geringerem fertigungstechnischem Aufwand werden in Proportionalventilen erreicht, die zunehmend als Stetigsteuerventile Verwendung finden. Sie sind dadurch gekennzeichnet, daß ein sog. Proportionalmagnet das elektrische Eingangssignal direkt in die annähernd proportionale Auslenkung eines Steuerschiebers umwandelt. Im einstufigen Ventil ist dies gleichzeitig der Hauptsteuerschieber, der i. allg. ebenfalls als Vierkantensteuerschieber ausgeführt ist. Im zweistufigen liegt wiederum, wie im Servoventil, der Hauptsteuerschieber in der Brückendiagonale des Vorsteuerschiebers. Bei der Modellierung des Servoventils kann wieder auf die Grundbeziehungen gemäß (3.9) zurückgegriffen werden. Aus Gründen des besseren Verständnisses soll mit der Hauptsteuerstufe, deren Eingangsgröße der Weg s_H ist, begonnen werden (Bild 3.11 a).

Die Hauptsteuerstufe ist über die Grundbeziehungen

$$\begin{aligned}
Q_1 &= B\,(s_{H0} - s_H)\sqrt{p_1} \\
Q_{01} &= B\,(s_{H0} + s_H)\sqrt{p_{DQ} - p_1} \\
Q_{02} &= B\,(s_{H0} - s_H)\sqrt{p_{DQ} - p_2} \\
Q_2 &= B\,(s_{H0} + s_H)\sqrt{p_2} \\
Q_V &= Q_{01} - Q_1 = Q_2 - Q_{02} \\
\Delta p &= p_1 - p_2 \\
B &= K_1 d\pi \quad \text{[s. Gl. (3.9 d)]}
\end{aligned} \tag{3.10}$$

zu beschreiben.

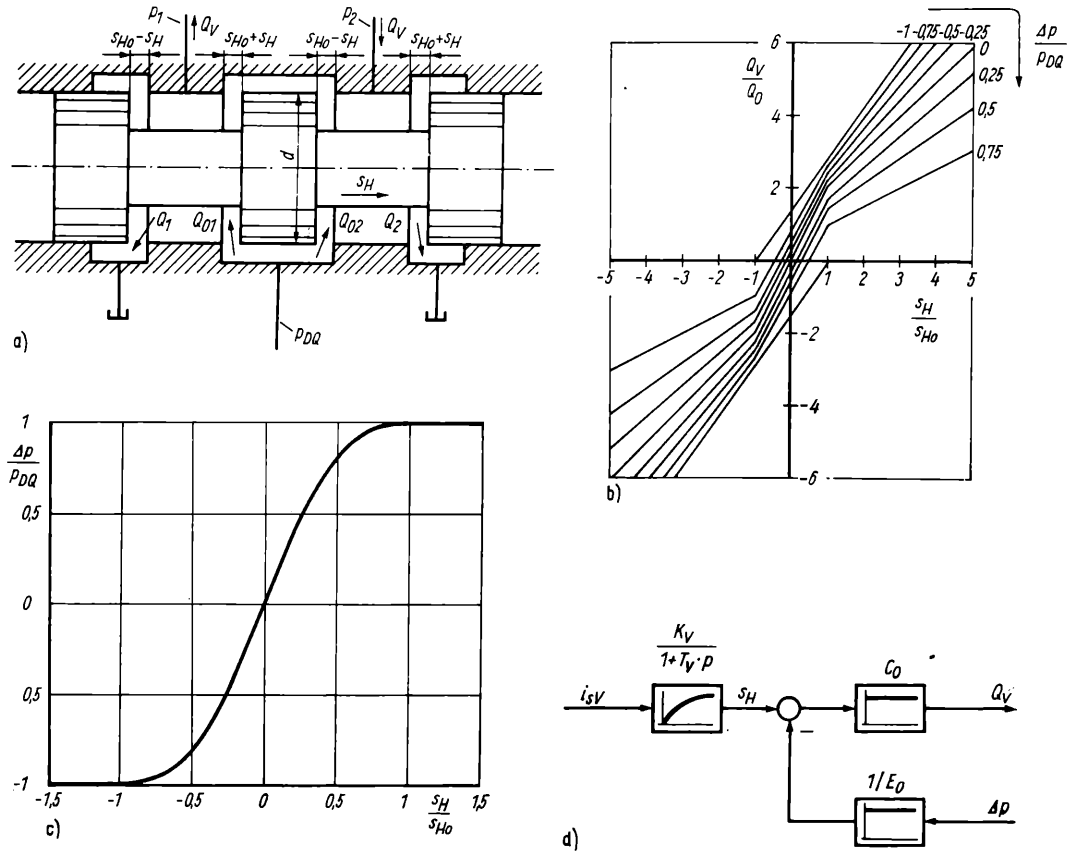


Bild 3.11. Hauptsteuerstufe eines Stetigsteuerventils (Servo- oder Proportionalventil mit Vierkantensteuerschieber)

a) Schema Hauptsteuerstufe; b) normiertes Kennlinienfeld $Q_V = f(s_H)$, Parameter; c) Leerlaufkennlinie, normiert

Aus den Beziehungen (3.10) ist nach mehreren Umrechnungen zu ermitteln:

1. die dem Kennlinienfeld (Bild 3.11 b) zugrunde gelegte allgemeine Beziehung

$$Q_V = B\,(s_{H0} + s_H)\sqrt{\frac{p_{DQ} - \Delta p}{2}} - B\,(s_{H0} - s_H)\sqrt{\frac{p_{DQ} + \Delta p}{2}} \tag{3.11}$$

(erster Term gleich Null bei $s_H \leqq -s_{H0}$, zweiter Term gleich Null bei $s_H \geqq s_{H0}$)

2. die Leerlaufkennlinie (Bild 3.11 c)

$$\Delta p = \begin{cases} 2p_{DQ}\dfrac{s_H/s_{H0}}{1 + (s_H/s_{H0})^2} & \text{für } |s_H| \leqq s_{H0} \\ +1 & \text{für } s_H > s_{H0} \\ -1 & \text{für } s_H < -s_{H0}. \end{cases} \tag{3.12}$$

Analog zu elektrischen Brückenschaltungen kann eine Linearisierung auch dieser hydraulischen Brücke an einem typischen Arbeitspunkt durchgeführt werden. Das ist bei Einsatz des Servoventils in einer Lageregelung die abgeglichene Brücke:

$$Q_V = 0, \quad s_H = 0.$$

Aus dem Anstieg der Leerlaufkennlinie im Nullpunkt ist die Druckverstärkung E_0, aus dem Anstieg der Kurzschlußkennlinie [s. Gl. (3.11) bei $\Delta p = 0$] die Ölstromverstärkung C_0 zu ermitteln. Es ergibt sich

$$E_0 = \left.\frac{\mathrm{d}\Delta p}{\mathrm{d}s_\mathrm{H}}\right|_{Q_\mathrm{V} = 0} = 2\,\frac{p_\mathrm{DQ}}{s_\mathrm{H0}} \tag{3.13}$$

$$C_0 = \left.\frac{\mathrm{d}Q_\mathrm{V}}{\mathrm{d}s_\mathrm{H}}\right|_{\Delta p = 0} = B\sqrt{2p_\mathrm{DQ}}\,. \tag{3.14}$$

Das führt auf die linearisierte Beziehung der Hauptsteuerstufe

$$s_\mathrm{H} = \frac{Q_\mathrm{V}}{C_0} + \frac{\Delta p}{E_0}. \tag{3.15}$$

Prinzipiell ähnliche Grundbeziehungen wie in der Hauptsteuerstufe gelten auch für die Vorsteuerstufe. Hinzu kommen das Kräftegleichgewicht am massebehafteten Steuerschieber als Belastung der Vorsteuerstufe, die Wirkung der Rückführfeder, das Kräftegleichgewicht an der ebenfalls massebehafteten Prallplatte und die Verzögerungen im Magnetsystem. Zwei stark verkoppelte Feder-Masse-Systeme erschweren die Modellierung über die Grundzusammenhänge. Untersuchungen, z. B. in [3.8], zeigen, daß bei Verwendung einer Ansteuerelektronik mit Stromausgang die Beschreibung der Vorsteuerstufe als proportional wirkendes Übertragungsglied mit Verzögerung 1. Ordnung, als sog. PT_1-Glied, ausreichend genau ist. Seine Übertragungsfunktion $G_\mathrm{V}(p)$ lautet dann:

$$G_\mathrm{V}(p) = \frac{s_\mathrm{H}(p)}{i_\mathrm{SV}(p)} = \frac{K_\mathrm{V}}{1 + T_\mathrm{V}p}; \tag{3.16}$$

K_V statischer Übertragungsfaktor der Vorsteuerstufe
T_V Zeitkonstante der Vorsteuerstufe, liegt im Bereich $T_\mathrm{V} = 1$ bis 5 ms.

An dieser Stelle taucht erstmals der Begriff Übertragungsfunktion $G(p)$ auf. Sie ist der Quotient der Laplace-Transformierten einer Ausgangs- und einer Eingangsgröße. Sie steht mit der Differentialgleichung eines linearen Systems in engem Zusammenhang. Beim Übergang vom Zeit- in den Bildbereich mit Hilfe der Laplace-Transformation wird die Ableitung nach der Zeit ($\mathrm{d}/\mathrm{d}t$) zum Laplace- oder Differentialoperator p, eine zweifache Ableitung ($\mathrm{d}^2/\mathrm{d}t^2$) zu p^2 usw. Ein Übertragungsglied mit der Differentialgleichung

$$T_3^3\dddot{x}_\mathrm{a} + T_2^2\ddot{x}_\mathrm{a} + T_1\dot{x}_\mathrm{a} + x_\mathrm{a} = K(x_\mathrm{e} + T'\dot{x}_\mathrm{e})$$

hat demnach die Übertragungsfunktion $G(p)$

$$G(p) = \frac{x_\mathrm{a}(p)}{x_\mathrm{e}(p)} = \frac{K(1 + T'p)}{1 + T_1p + T_2^2p^2 + T_3^3p^3}.$$

Die Übertragungsfunktion (3.16) entspricht im Zeitbereich der Differentialgleichung

$$T_\mathrm{V}\dot{s}_\mathrm{H} + s_\mathrm{H} = K_\mathrm{V}i_\mathrm{SV}.$$

Die Gln. (3.15) und (3.16) führen auf das vereinfachte, linearisierte Signalflußbild des Servoventils nach Bild 3.11d, wie es für die Ermittlung der Kennwerte der Antriebe zugrunde gelegt wird.

Von Servoventilen bekannte Zahlenwerte der Hauptparameter liegen bei

$$E_{0i} = E_0 K_V = (1 \text{ bis } 3)\ \mathrm{mA}^{-1} \cdot p_{DQ}$$

und

$$C_{0i} = C_0 K_V = (1 \text{ bis } 3)\ \frac{\mathrm{cm}^3}{\mathrm{s} \cdot \mathrm{mA} \cdot \sqrt{\mathrm{MPa}}} \cdot \sqrt{p_{DQ}}\ .$$

3.3.1.3. Übertragungsverhalten und Kennwerte der Antriebe

Gemäß den im Abschnitt 3.1. formulierten Voraussetzungen ist für den Antrieb, bestehend aus Steuerventil und Verbraucher, die Abhängigkeit seines Bewegungszustands von der auf ihn wirkenden reduzierten Kraft F_L und vor allem vom Eingangsstrom i_{SV} des Steuerventils von Interesse.

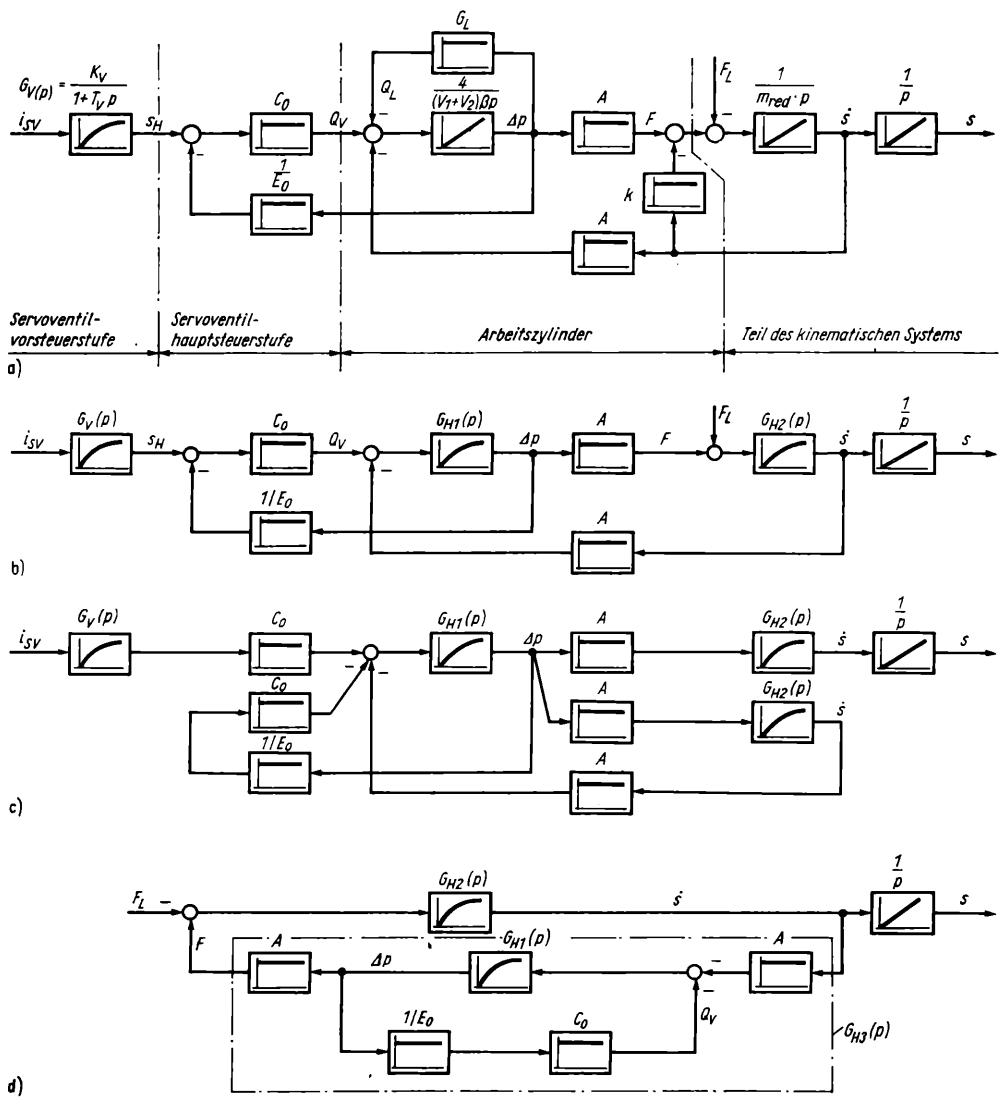


Bild 3.12. Signalflußbild (linearisiert) der Funktionsgruppe Servoventil-Druckstromverbraucher
a) ausführliche Darstellung; b) bis d) Stufen der Zusammenfassung

Servoventil und Verbraucher müssen infolge der Rückwirkung des Verbrauchers auf das Servoventil als Einheit betrachtet werden. Im Bild 3.12a ist ein Signalflußbild dieser Funktionsgruppe dargestellt. Gewählt wurde – wie meist in Industrieroboterantrieben angewendet – ein Arbeitszylinder mit beiderseitiger Kolbenstange. Bild 3.12a ist die Zusammenführung der Bilder 3.9c und 3.11d; die Reibung wurde geschwindigkeitsproportional angesetzt. Da damit das Signalflußbild linear ist, können Übertragungsfunktionen $G(p)$ des Systems ermittelt werden.

Nach Bild 3.12a sind dies:

$$s(p)/i_{SV}(p) \quad \text{und} \quad s(p)/F_L(p).$$

Ihre Berechnung wird aufwendig, wenn z.B. aus den im Signalflußbild dargestellten Grundbeziehungen und Bilanzgleichungen durch Eliminieren der Zwischengrößen die gesuchte Abhängigkeit ermittelt wird. Der Regelungstechniker geht anders vor: Einfache Wirkungsschleifen werden zuerst berechnet und sich kreuzende Rückwirkungen so umgezeichnet, daß alle an derselben Stelle beginnen und an derselben Bilanz- (Summations-) Stelle enden. Die gewünschte Übertragungsfunktion ist dann jeweils über den einfachen Ansatz

$$G(p) = \frac{G_{\text{vorwärts}}(p)}{1 + G_{\text{Kreis}}(p)} \tag{3.17}$$

zu berechnen, wie an den folgenden Beispielen gezeigt wird. Werden die einfachen Wirkungsschleifen im Bild 3.12a infolge der Reibkraft (k) und des Leckleitwerts (G_L) gemäß Ansatz (3.17) berechnet zu

$$G_{H1}(p) = \frac{\dfrac{4}{(V_1 + V_2)\,\beta p}}{1 + \dfrac{4}{(V_1 + V_2)\beta p} G_L} = \frac{1}{G_L + (V_1 + V_2)\dfrac{\beta}{4} p}$$

$$G_{H2}(p) = \frac{\dfrac{1}{m_{red} p}}{1 + \dfrac{1}{m_{red} p} k} = \frac{1}{k + m_{red} p},$$

so ergibt sich aus Bild 3.12a das Bild 3.12b.

Zur Berechnung der Übertragungsfunktionen $s(p)/i_{SV}(p)$ und $s(p)/F_L(p)$ wurde Bild 3.12b in die Bilder 3.12c und 3.12d umgeformt.

Im ersten Fall ist

$$G_{\text{vorwärts}}(p) = \frac{K_V}{1 + T_V p} C_0 G_{H1}(p)\, A G_{H2}(p) \frac{1}{p}$$

$$G_{\text{Kreis}}(p) = G_{H1}(p) \left(A^2 G_{H2}(p) + \frac{C_0}{E_0} \right).$$

Im zweiten Fall entsteht eine zusätzliche Rückführung, so daß über

$$G_{H3}(p) = \frac{A^2 G_{H1}(p)}{1 + G_{H1}(p) \dfrac{C_0}{E_0}}$$

anzusetzen sind

$$G_{\text{vorwärts}}(p) = G_{\text{H2}}(p)\,\frac{1}{p}$$

$$G_{\text{Kreis}}(p) = G_{\text{H2}}(p)\,G_{\text{H3}}(p).$$

Werden diese Beziehungen in Gl. (3.17) eingesetzt, so ergeben sich nach mehreren Zwischenschritten die gesuchten Übertragungsfunktionen. Ihre Form wird so gewählt, daß die typischen Kennwerte Eigenzeitkonstante T, Dämpfung D, Nachlaufzeitkonstante T_{N}, Vorhaltzeitkonstante T'_{V} und der statische Übertragungsfaktor K_F zwischen Kraft F_{L} und Geschwindigkeit s explizit ablesbar sind.

Es ergibt sich

$$\frac{s_p}{i_{\text{SV}}(p)} = \frac{1}{T_{\text{N}p}\,(1 + 2DTp + T^2p^2)}\,\frac{1}{1 + T_{\text{V}}p} \tag{3.18}$$

$$\frac{s(p)}{F_{\text{L}}(p)} = -\frac{K_F}{p}\,\frac{1 + T'_{\text{V}}p}{1 + 2DTp + T^2p^2} \tag{3.19}$$

$$T_{\text{N}} = \frac{A^2 + k\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right)}{C_{0i}A} \tag{3.20}$$

$$T = \sqrt{\frac{(V_1 + V_2)\,\dfrac{\beta}{4}\,m_{\text{red}}}{A^2 + K\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right)}} \tag{3.21}$$

$$D = \frac{m\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right) + k\,(V_1 + V_2)\,\dfrac{\beta}{4}}{2\sqrt{\left(A^2 + k\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right)\right)(V_1 + V_2)\,\dfrac{\beta}{4}\,m_{\text{red}}}} \tag{3.22}$$

$$2DT = \frac{m_{\text{red}}\,\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}} + k\,(V_1 + V_2)\,\dfrac{\beta}{4}}{A^2 + k\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right)} \tag{3.23}$$

$$K_F = \frac{\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}}{A^2 + k\left(\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}\right)} \tag{3.24}$$

$$T'_{\text{V}} = \frac{(V_1 + V_2)\,\dfrac{\beta}{4}}{\dfrac{C_{0i}}{E_{0i}} + G_{\text{L}}}. \tag{3.25}$$

Beachte: $C_{0i} = C_0K_{\text{V}}$, $E_{0i} = E_0K_{\text{V}}$.

Zu diesen Beziehungen können folgende Hinweise gegeben werden:

- In der Summe $A^2 + k(C_{0i}/E_{0i} + G_L)$ ist i. allg. A^2 der dominierende Summand [s. Gln. (3.20) bis (3.24)].
- Die Eigenzeitkonstante T ist trotz großer bewegter reduzierter Massen m_{red} infolge kleiner Volumina relativ klein. Mit z. B. $m_{red} = 1000$ bzw. 10000 kg, $V_1 = V_2 = 100\ cm^3$ und $A = 10\ cm^2$ ergibt sich $T \approx 6{,}3$ ms bzw. 20 ms.
- Dämpfend wirken zwei Einflüsse: die Reibung und das innere Lecköl. Beide sind in vielen Fällen so gering, daß sich sehr kleine Dämpfungswerte ($D < 0{,}01$) ergeben, was sich ungünstig auf das dynamische Verhalten auswirkt. Aber künstliche Vergrößerung der Reibung oder des Lecköls haben innerhalb der Lageregelung negativen Einfluß auf die Positioniergenauigkeit.
- Bei Einsatz eines rotatorischen Verbrauchers, etwa eines Hydromotors, reduziert sich das Volumen $V_1 + V_2$ auf die Summe der Leitungsvolumina zwischen Hydromotor und Servoventil.
- Verfälschen bestimmte Linearisierungen vor allem im Servoventilmodell bei simulierten Einsatzfällen des Antriebs das tatsächliche Verhalten zu stark, dann sind diese zumindest teilweise rückgängig zu machen. Beispiele dafür werden im Abschnitt 4. beschrieben.

3.3.1.4. Statisches und dynamisches Verhalten der Druckquelle

Als Funktionseinheit Druckquelle kommt für Industrieroboter vor allem die Verstellpumpe mit sog. Nullhubregelung zum Einsatz (Schema s. Bild 3.13a). Die sehr gute statische Kennlinie dieser Druckregelung (Bild 3.13b) ist verknüpft mit dem Vorteil, daß die Verstellpumpe nur soviel fördert, wie die Verbraucher aktuell fordern. Der statische Fehler von $\Delta p_{stat} = (0{,}02 \text{ bis } 0{,}1)\, p_{DQ}$ im Bereich der Pumpenfördermenge $0 < Q_p < Q_{p\,max}$ ist für die Anforderungen der Industrieroboter zulässig. Durch das dynamische Verhalten verursachte Druckschwankungen können wesentlich größer sein; Möglichkeiten ihrer Beeinflussung werden im folgenden angegeben.

Der Druckregelkreis kann in erster Näherung wie im Bild 3.13c dargestellt beschrieben werden. Die Hauptzeitkonstante der Regelstrecke T_s wird durch den Gesamtwiderstand W aller Verbraucher (minimal: $W_{min} \approx p_{DQ}/Q_{p\,max}$) und die Kapazität C_y des Leitungssystems zwischen Druckquelle und Verbraucherschaltungen einschließlich angeschlossener Druckflüssigkeitsspeicher bestimmt (s. auch (3.9) und [3.2]):

$$T_s = WC_y \tag{3.26}$$

$$W \approx \frac{p_{DQ}}{Q_{p\,max}} \tag{3.27}$$

$$C_y \approx V_L\beta + \sum_i \left(V_{Ni} \frac{p_{gas\,i}}{p_{DQ0}^2} \right); \tag{3.28}$$

V_L Volumen des Leitungssystems
β Kompressibilität des Öls ($\beta \approx 8 \cdot 10^{-4}\ MPa^{-1}$)
V_N Nennvolumen eines Druckflüssigkeitsspeichers
p_{gas} Gasfülldruck des Speichers
P_{DQ0} Arbeitspunkt des Druckes.

Die Hauptzeitkonstante der Regeleinrichtung T_R liegt aus Stabilitäts- und ökonomischen Gründen (hochohmige Vorsteuerstufe) vielfach bei $T_R \approx (1 \text{ bis } 3)$ s.

Bibliothek

Da diese Baugruppe vor allem infolge der Vorsteuerung ein sehr kompliziertes Modell besitzt, kann T_R hier nicht ausführlich hergeleitet werden.

Der statische Übertragungsfaktor K_R ist näherungsweise aus der statischen Kennlinie zu ermitteln:

$$K_R = \frac{\Delta Q}{\Delta p_{DQ}} \frac{Q_{p\,max}}{(0{,}02 \text{ bis } 0{,}1)\, p_{DQ}}. \tag{3.29}$$

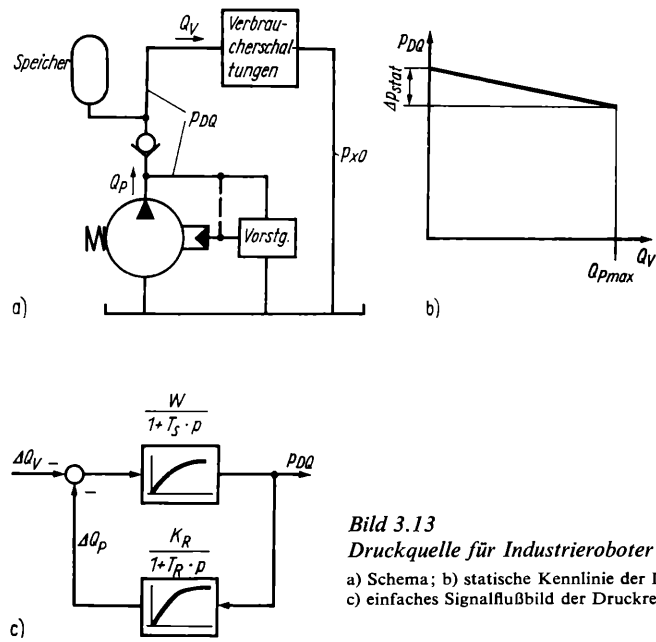


Bild 3.13
Druckquelle für Industrieroboter
a) Schema; b) statische Kennlinie der Druckregelung; c) einfaches Signalflußbild der Druckregelung

Die Kreisverstärkung der Regelung ist

$$V_0 = K_R W = \frac{p_{DQ}}{Q_p} \frac{Q_{p\,max}}{(0{,}02 \text{ bis } 0{,}1)\, p_{DQ}} = (10 \text{ bis } 50) \frac{Q_{p\,max}}{Q_p}. \tag{3.30}$$

Stehen alle Verbraucher still, dann hat die Kreisverstärkung wegen $W \to W_{max}$ bzw. $Q_p \to Q_{min}$ ihren maximalen Wert. Für $W \to \infty$ geht die Übertragungsfunktion der Regelstrecke in die Form $1/C_y p$ über. Die Störübertragungsfunktion $G_Z(p)$ lautet dann wie folgt [s. auch (3.17)]:

$$G_Z(p) = \frac{\Delta p_{DQ}(p)}{\Delta Q_V(p)} = \frac{\dfrac{1}{C_y p}}{1 + \dfrac{1}{C_y p} \dfrac{K_R}{1 + T_R p}} = \frac{1}{K_R} \frac{1 + T_R p}{1 + \dfrac{C_y}{K_R} p + \dfrac{C_y T_R}{K_R} p}$$

$$= \frac{1}{K_R} \frac{1 + T_R p}{1 + 2 D_D T_D p + T_D^2 p^2}. \tag{3.31}$$

Die Dämpfung D_D der Druckregelung ist

$$D_D = \frac{1}{2} \sqrt{\frac{C_y}{K_R T_R}}. \tag{3.32}$$

Diese Beziehung sagt aus, daß eine ausreichend große Kapazität des Leitungssystems geschaffen werden muß, um auch an ungünstigen Arbeitspunkten eine ausreichend bedämpfte Druckregelung zu sichern. Wird das ungenügend beachtet, können sehr große gegenseitige Beeinflussungen der Antriebe entstehen.

Diese treten natürlich auch auf, wenn die Druckquelle nicht groß genug dimensioniert ist und der Druck bei schnellen Bewegungen zusammenbricht.

3.3.2. Elektrische Antriebe für Lageregelungen

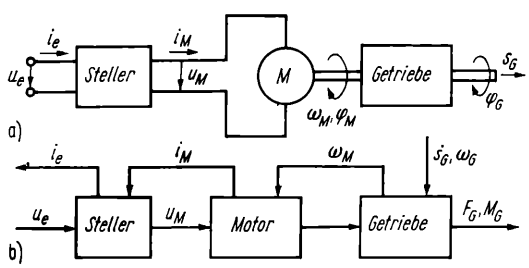
3.3.2.1. Struktur

In elektrischen Antrieben für Lageregelungen werden heute vorwiegend Gleichstrommotoren, die von an die Motorparameter angepaßten Thyristor- oder Transistorstellgliedern angesteuert werden, eingesetzt. Im Gegensatz zu den hydraulischen Antrieben kommen in diesen Antrieben hochübersetzende Getriebe zum Einsatz, da die vom Elektromotor zu realisierenden Momente für einen Direktantrieb zu klein sind.

Die notwendige Anpassung von Motor und Stellglied hat dazu geführt, daß beide i. allg. als Einheit angeboten werden. Die erforderlichen Überwachungs- und Sicherheitsschaltungen haben auf die normale Funktion und damit auf das zu ermittelnde Modell keinen Einfluß.

Anders ist das bei den in solchen Antrieben oft vorhandenen Ankerstromregelungen und – als Meßglied dient der in diesem Abschnitt noch zu beschreibende Tachogenerator – Drehzahlregelungen. Diese Regelungen sind jedoch kein Standard. In vielen Antrieben wird z. B. eine einfache Ankerstrombegrenzung, die nur bei hohen Beschleunigungen des Antriebs wirksam wird, anstelle der Ankerstromregelung realisiert; in Industrierobotern wird die der Lageregelung unterlagerte Geschwindigkeitsregelung oft in der Bewegungssteuerung untergebracht (s. Abschn. 4.).

Im Rahmen dieses Abschnitts soll in Anlehnung an den Abschnitt 3.3.1. als zu beschreibender Antrieb die Übertragungskette Stellglied–Motor–Getriebe analysiert werden.



a) Schema
b) qualitatives Signalflußbild

Bild 3.14. Struktur eines elektrischen Industrieroboterantriebs für Lageregelungen, bestehend aus Steller, Motor und Getriebe

Im Bild 3.14 sind Schema und qualitatives Signalflußbild des elektrischen Antriebs für Lageregelungen dargestellt. Im Signalflußbild ist der Strom i_e nur dann von Interesse, wenn durch den Steller eine Rückwirkung auf den informationsverarbeitenden Teil des Regelkreises oder auf eine Vorverstärkerstufe eintritt. Das soll im folgenden nicht der Fall sein.

3.3.2.2. Bestandteile der elektrischen Antriebe und ihre Modelle

Als wesentliche Grundbeziehungen für die Modellierung des dynamischen Verhaltens elektrischer und elektromechanischer Baugruppen sind anzusetzen:

- die Strom-Spannungs-Beziehungen von ohmschem Widerstand R, Induktivität L und Kapazität C

$$u_R = Ri_R \quad \text{bzw.} \quad \frac{u_R(p)}{i_R(p)} = R$$

$$u_L = L\frac{\mathrm{d}i_L}{\mathrm{d}t} \quad \text{bzw.} \quad \frac{u_L(p)}{i_L(p)} = pL \tag{3.33}$$

$$u_C = \frac{1}{C}\int i_C \mathrm{d}t \quad \text{bzw.} \quad \frac{u_C(p)}{i_C(p)} = \frac{1}{pC}$$

- das Induktionsgesetz:

  Eine elektrische Spannung e wird erzeugt, wenn sich ein Leiter durch ein Magnetfeld bewegt. Im Elektromotor gilt

$$e = K_M \Phi \omega_M; \tag{3.34}$$

K_M Motorkonstante
Φ magnetischer Fluß
ω_M Rotorwinkelgeschwindigkeit

- das Kraftgesetz:

  Das auf eine stromführende Spule ausgeübte Drehmoment M'_M ist der Stromstärke i_M und dem von der Spule umfaßten magnetischen Fluß Φ proportional. Im Elektromotor ist anzusetzen [K_M ist dieselbe Motorkonstante wie in (3.34)]:

$$M'_M = K_M \Phi i_M. \tag{3.35}$$

Weitere Grundbeziehungen sind dem Abschnitt 3.3.1. zu entnehmen.

Stellmotor

Die Industrierobotertechnik stellt hohe Anforderungen an das Massenträgheitsmoment-Drehmoment-Verhältnis und an das Masse-Drehmoment-Verhältnis des Motors, da der Motor teilweise mitbewegt werden muß. Es wurden vor allem permanenterregte Gleichstrommotoren für diese Aufgabe entwickelt und eingesetzt [3.5]. Im Bild 3.15a ist die Ersatzschaltung eines solchen permanenterregten Gleichstrommotors abgebildet, im Bild 3.15b das zugehörige linearisierte Signalflußbild. Die darin vorkommenden Bilanzgleichungen sind die Spannungsbilanz

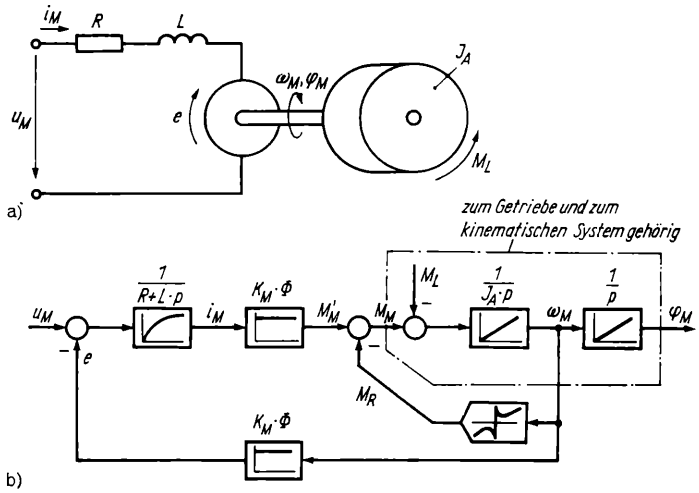
$$u_M - e = iR + L\frac{\mathrm{d}i_M}{\mathrm{d}t} \tag{3.36}$$

und die Momentenbilanz, bestehend aus Beschleunigungsmoment $M_B = J_A \ddot{\varphi}_M$, innerem Motormoment M'_M, Reibmoment M_R und äußerem Belastungsmoment M_L,

$$J_A \cdot \ddot{\varphi}_M = M'_M - M_R - M_L. \tag{3.37}$$

Das enthaltene Massenträgheitsmoment ist im Einsatzfall den bewegten Massen des kinematischen Systems zuzuschlagen. Für die Definition motorspezifischer Größen ist das Ankerträgheitsmoment J_A anzusetzen.

Wird das Reibmoment M_R vorerst vernachlässigt, so ist das Signalflußbild einschleifig. Das ist für den Motor selbst, d.h. ohne Getriebe, meist erlaubt, da diese Rückwirkung gegenüber der induzierten Spannung e klein ist.



	Gleichstromnebenschlußmotoren			
	mit Erregerwicklung	mit Permanentmagneten		
		Universalbauform	mit Scheibenläufer	mit Schlankanker
Mech. Leistung	10 ... 320 W	1 ... 320 W	25 W ... 5 kW	100 W ... 2 kW
Wirkungsgrad	0,3 ... 0,75	0,25 ... 0,8	0,4 ... 0,85	0,5 ... 0,8
Stromüberlastbarkeit	2	2	8	5
T_A	10 ... 25 ms	5 ... 30 ms	0,2 ... 0,5 ms	2 ... 4 ms
T_M	30 ... 40 ms	20 ... 30 ms	5 ... 20 ms	5 ... 8 ms

c)

Bild 3.15. Gleichstromnebenschlußmotor mit Spannungssteuerung

a) Schema eines permanent erregten Gleichstrommotors; b) Signalflußbild dieses Motors; c) Kennwerte für verschiedene spannungsgesteuerte Gleichstromnebenschlußmotoren (nach [3.2])

Die Ansätze für die gesuchten Übertragungsfunktionen des Motors $M(p)/u_M(p)$ und $M(p)/M_L(p)$ sind gemäß Gl. (3.17)

$$\frac{\varphi_M(p)}{u_M(p)} = \frac{\dfrac{1}{R+Lp} K_M\Phi \dfrac{1}{J_A p} \dfrac{1}{p}}{1 + \dfrac{1}{R+Lp} (K_M\Phi)^2 \dfrac{1}{J_A p}}$$

bzw.

$$\frac{\varphi_M(p)}{M_L(p)} = \frac{\dfrac{1}{J_A p} \dfrac{1}{p}}{1 + \dfrac{1}{R+Lp} (K_M\Phi)^2 \dfrac{1}{J_A p}}.$$

Unter Nutzung der Beziehungen für die Ankerzeitkonstante T_A und die elektromechanische Zeitkonstante T_M

$$T_A = \frac{L}{R} \tag{3.38}$$

$$T_M = \frac{J_A R}{(K_M \Phi)^2} \tag{3.39}$$

ergibt sich nach mehreren Rechenschritten

$$\frac{\varphi_M(p)}{u_M(p)} = \frac{1}{pK_M\Phi} \frac{1}{1 + T_M p(1 + T_A p)} \tag{3.40}$$

$$\frac{\varphi_M(p)}{M_L(p)} = -\frac{R}{p(K_M\Phi)^2} \frac{1 + T_A p}{1 + T_M p(1 + T_A p)}. \tag{3.41}$$

Im Bild 3.15c sind Zeitkonstanten und einige weitere Kennwerte von Gleichstrom-Kleinmotoren aufgeführt.

Stellglied

Stellglieder ermöglichen durch Eingriff in den Energiefluß vom Netz zum Motor die Steuerung der Motorspannung hinsichtlich Vorzeichen (mit Umkehrschaltungen) und Betrag (mit Hilfe von Stromrichterschaltungen). In Antrieben für Industrieroboter ist das Stellglied eine vollelektronische Baugruppe, deren Hauptbauelemente Thyristoren, Leistungstransistoren oder Triacs sind. Die Ausgangsspannung ist i. allg. eine pulsierende Gleichspannung, deren Mittelwert die maßgebende Eingangsgröße für den Motor ist.

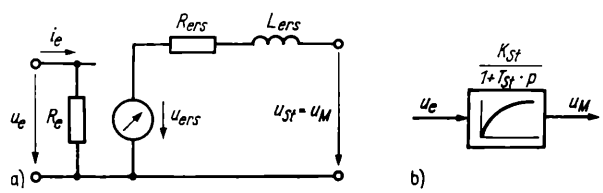


Bild 3.16
Steller
a) Ersatzschaltbild; b) Übertragungsverhalten

Im Bild 3.16a ist die übliche Ersatzschaltung [3.3] für derartige Steller angegeben. Die Ersatzspannung u_{ers} ist die von der Eingangsspannung u_e gesteuerte Ausgangsspannung des unbelasteten Stellers. Der Ersatzwiderstand R_{ers} berücksichtigt die Gesamtheit der vom Laststrom abhängigen Verluste im Stellglied, die Ersatzinduktivität L_{ers}, die Gesamtwirkung von Glättungsdrosseln und im Stellglied vorhandenen Streuinduktivitäten.

Die laststromabhängigen Verluste und die Streuinduktivitäten führen im Steller zu mehreren kleineren Signalverzögerungen, die ohne merklichen Fehler zu einer Summenzeitkonstante des Stellglieds T_{St} zusammengefaßt werden können. Oft wird das dynamische Verhalten vor allem der Stromrichter als mit einer Totzeit behaftet beschrieben. Wie Untersuchungen des Kleinsignalverhaltens ergaben, spiegelt sich dieser Ansatz im tatsächlichen Verhalten nicht wider [3.3].

Der Eingangswiderstand R_e ist i. allg. so groß, daß der Stellereingang als rückwirkungsfrei angesehen werden kann. Werden R_{ers} und L_{ers} dem Ankerkreiswiderstand R bzw. der Ankerkreisinduktivität L zugeschlagen (es ist dann $u_{ers} = u_{St} = u_M$ zu setzen), kann der

Steller bei der Verarbeitung ausreichend kleiner Eingangssignale analog Gl. (3.16) als PT_1-Glied beschrieben werden:

$$\frac{u_M(p)}{u_e(p)} = \frac{K_{St}}{1 + T_{St}p}; \tag{3.42}$$

K_{St} statischer Übertragungsfaktor des Stellglieds
T_{St} Verzögerungszeitkonstante des Stellglieds.

Bei zu großen Eingangssignalen ist die Begrenzung der Ausgangsspannung des Stellers zu beachten.

Der Signalfluß ist im Bild 3.16b dargestellt.

Getriebe

Bei exakter Beschreibung der Getriebe sind neben der seiner Aufgabe entsprechenden Bewegungs- und damit Kraft- bzw. Momentenumformung vor allem folgende Einflüsse im Modell zu berücksichtigen:

- die Torsionsnachgiebigkeit der antreibenden Welle
- die Massenträgheit der Bauelemente des Getriebes
- die Reibung
- das Getriebespiel.

In Industrierobotern sind die zwischen Motor und Kinematiksystem auftretenden Verformungen meist klein gegenüber denen im kinematischen System selbst. In den Lageregelungen finden weitgehend spielfreie Baugruppen Anwendung. Werden zusätzlich die Massenträgheit dem kinematischen System und die Reibungseinflüsse denen des Motors zugeschlagen, genügt zur Beschreibung der Getriebe i. allg. das Übersetzungsverhältnis bzw. der Umsetzungsfaktor.

Das in Industrierobotern mit lagegeregelten elektrischen Antrieben am häufigsten angewendeten Übersetzungsgetriebe (von Drehbewegung in Drehbewegung) ist das Wellgetriebe (harmonic drive). Durch sein hohes Übersetzungsverhältnis, durch geringen Raumbedarf und geringe Eigenmasse ist es für Industrieroboter besonders geeignet.

Es werden aber auch Schneckengetriebe, mit denen Übersetzungsverhältnisse bis zu 50 erreichbar sind, angewendet. Sie sind einfacher zu fertigen, haben aber große Reibung. Elektrische Antriebe, die eine translatorische Abtriebsbewegung erzeugen, besitzen ein Umsetzungsgetriebe von Drehbewegung auf lineare Bewegung. Dafür sind die aus den Werkzeugmaschinen bekannten Wälzschraubtriebe in den Industrierobotern am häufigsten anzutreffen; es kommen aber auch Ritzel-Zahnstange-Paarungen zum Einsatz.

In den Bildern 3.17a und b sind Schnitt und Wirkungsschema einer Variante des Wellgetriebes dargestellt. Das flexible Umlaufrad wird vom ovalen Wellgenerator mit seiner Zahnung in die des Mittelrads gedrückt. Enthält das Mittelrad (kreisförmig) z. B. $z_{MR} = 102$ und das Umlaufrad $z_{UR} = 100$ Zähne (die Teilellipse hat einen geringeren Umfang als der Teilkreis; das Umfangsverhältnis muß hier 100:102 sein), so ergibt sich über (s. [3.1])

$$i_1 = \frac{\omega_M}{\omega_G} = \frac{z_{UR}}{-z_{MR} + z_{UR}} \tag{3.43a}$$

eine Übersetzung i_1 von

$$i_1 = \frac{100}{-102 + 100} = -50.$$

Der Index 1 steht für das Übersetzungsgetriebe. Die Abtriebsdrehrichtung ist der des Wellgenerators entgegengesetzt ($i_1 < 0$), die Abtriebsdrehzahl nur $^1/_{50}$ der des Antriebs.

Ferner sei zum Wellgetriebe angemerkt, daß die Differenz $\Delta z = z_{MR} - z_{UR}$ der beiden Zähnezahlen geradzahlig sein muß. Um das angestrebte große Übersetzungsverhältnis i_1 zu erreichen, wird das Minimum $\Delta z = 2$ meist anzutreffen sein (s. Bild 3.17b). Dabei weicht die Teilellipse nur wenig vom Kreis ab, und die Zahnhöhen sind relativ gering, was zum einen die übertragbaren Momente beschränkt, zum anderen aber nur geringe Verformungen des Abtriebsrads verursacht.

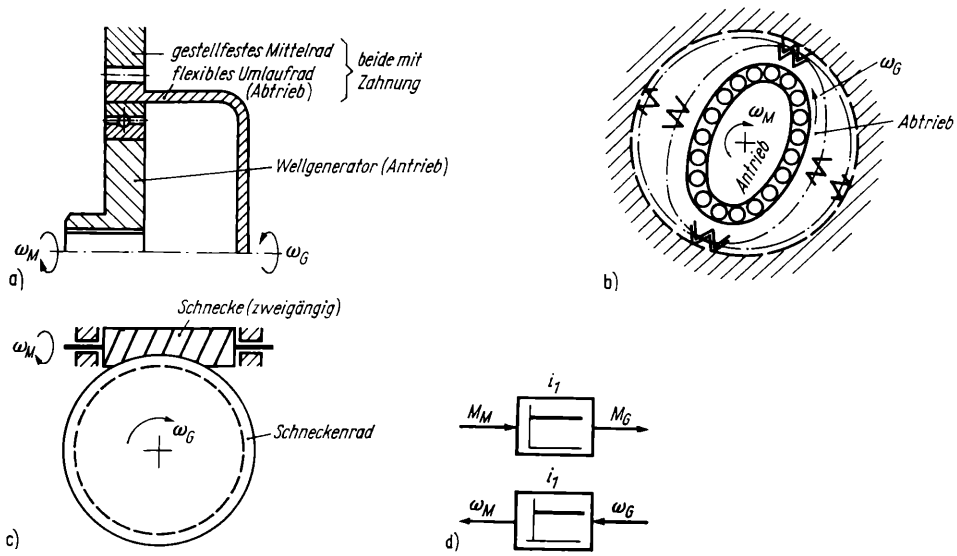


Bild 3.17. Übersetzungsgetriebe

a), b) Schnittbild und Wirkungsschema eines Wellgetriebes; c) Wirkungsschema eines Schneckengetriebes; d) Übertragungsverhalten eines Übersetzungsgetriebes

Im Bild 3.17 c ist das Schema eines Schneckengetriebes dargestellt. Die Schnecke ist als ein- oder mehrgängige Schraube ausgebildet (Gangzahl z_g). Das Übersetzungsverhältnis ermittelt sich für die Zähnezahl z des Schneckenrads zu

$$i_1 = \frac{\omega_M}{\omega_G} = \frac{z_g}{z}. \tag{3.43b}$$

Das Verhältnis des theoretischen Abtriebsmoments zum Motormoment ist ebenfalls gleich dem Übersetzungsverhältnis. Werden voraussetzungsgemäß die (infolge der hohen Übersetzung meist vernachlässigbare) Massenträgheit des Getriebes dem kinematischen System und die Reibungseinflüsse dem Motor zugeschlagen, so kann für das Modell des Übersetzungsgetriebes allgemein

$$\begin{aligned} M_G &= i_1 M_M \\ \omega_M &= i_1 \omega_G \end{aligned} \tag{3.44}$$

angesetzt werden (Bild 3.17d).

In den Bildern 3.18a und b sind die Schemata üblicher Umsetzungsgetriebe abgebildet. Für ihren Umsetzungsfaktor i_2 sind die entsprechenden Beziehungen einfach aus diesen Bildern zu entnehmen.

Für den Schraubtrieb gilt

$$i_2 = \frac{\omega_G}{\dot{s}_G} = \frac{F_G}{M_G} = \frac{2\pi}{h}; \tag{3.45a}$$

h Gewindesteigung je Umdrehung.

Für Ritzel–Zahnstange gilt

$$i_2 = = \frac{\omega_G}{\dot{s}_G} = \frac{F_G}{M_G} = \frac{2}{d_0} = \frac{2}{mz}; \tag{3.45b}$$

d_0 Teilkreisdurchmesser des Ritzels
m Zahnmodul
z Zähnezahl des Ritzels.

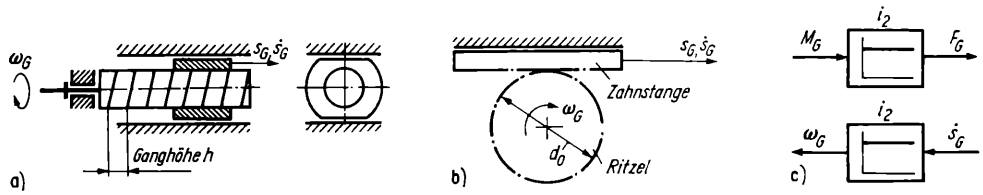


Bild 3.18. Umsetzungsgetriebe von Rotation auf Translation
a), b) Wirkungsschema eines Schraubtriebs und einer Ritzel-Zahnstange-Paarung; c) Übertragungsverhalten eines Umsetzungsgetriebes

Im Bild 3.18c ist wiederum das Übertragungsverhalten dargestellt (Massenträgheits- und Reibungseinflüsse ebenfalls dem Motor zugeschlagen):

$$\begin{aligned} F_G &= i_2 M_G \\ \omega_G &= i_2 \dot{s}_G . \end{aligned} \tag{3.46}$$

3.3.2.3. Übertragungsverhalten und Kennwerte des elektrischen Antriebs

Die für die drei Bestandteile des Antriebs abgeleiteten Modelle sind im Bild 3.19 zum Signalflußbild des gesamten Antriebs zusammengefaßt worden. Ein Vergleich mit Bild 3.12a zeigt, daß die Struktur etwas einfacher ist: es existieren keine sich kreuzenden Rückwirkungen.

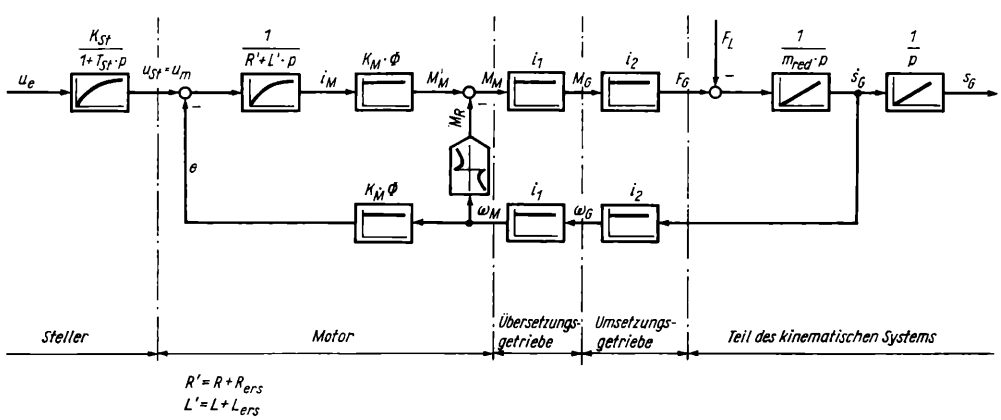


Bild 3.19. Detailliertes Signalflußbild der Funktionsgruppe Steller–Motor–Übersetzungsgetriebe–Umsetzungsgetriebe

Bei der Ermittlung der Übertragungsfunktionen $s_G(p)/u_e(p)$ und $s_G(p)/F_L(p)$ soll zunächst wiederum die Reibkraft vernachlässigt werden. Auf das dann einschleifige Signalflußbild ist Gl. (3.17) sofort anwendbar.

Aus den Ansätzen

$$\frac{s_G(p)}{u_e(p)} = \frac{\dfrac{K_{St}}{1+T_{St}p}\,\dfrac{1}{R'+L'p}\,K_M\Phi i_1 i_2\,\dfrac{1}{m_{red}p}\,\dfrac{1}{p}}{1+\dfrac{1}{R'+L'p}\,(K_M\Phi i_1 i_2)^2\,\dfrac{1}{m_{red}p}}$$

und

$$\frac{s_G(p)}{F_L(p)} = \frac{\dfrac{1}{m_{red}p}\,\dfrac{1}{p}}{1+\dfrac{1}{R'+L'p}\,(K_M\Phi i_1 i_2)^2\,\dfrac{1}{m_{red}p}}$$

sind die Übertragungsfunktionen

$$\frac{s_G(p)}{u_e(p)} = \frac{K_{St}}{pK_M\Phi i_1 i_2}\,\frac{1}{1+T'_M p\,(1+T'_A p)}\,\frac{1}{1+T_{St}p} \tag{3.47}$$

$$\frac{s_G(p)}{F_L(p)} = -\frac{R+R_{ers}}{p\,(K_M\Phi i_1 i_2)^2}\,\frac{1+T'_A p}{1+T'_M p\,(1+T'_A p)} \tag{3.48}$$

$$T'_A = \frac{L+L_{ers}}{R+R_{ers}} = \frac{L'}{R'} \tag{3.49}$$

$$T'_M = \frac{m_{red}\,(R+R_{ers})}{(K_M\Phi i_1 i_2)^2} = \frac{m_{red}R'}{(K_M\Phi i_1 i_2)^2} \tag{3.50}$$

zu ermitteln. Ein Vergleich mit den ähnlichen Beziehungen des Motors allein [s. Gln. (3.38) bis (3.41)] zeigt, daß die Ankerzeitkonstante T'_A vom Steller nur wenig beeinflußt wird, daß sich aber die elektromechanische Zeitkonstante T'_M durch das Getriebe sehr stark verringern kann. Das trifft auch auf die statischen Übertragungsfaktoren zu. Günstig ist dies bei der Rückwirkung der äußeren Kräfte; ungünstig ist die relativ geringe erreichbare Stellgeschwindigkeit s_G, die sich bei großen Übersetzungsverhältnissen i_1, i_2 ergibt.

Nicht so einfach zu überschauende Zusammenhänge existieren bei der Berücksichtigung der Reibung. Ist das Reibmoment M_R eine nichtlineare Funktion der Winkelgeschwindigkeit ω_M, wie im entsprechenden Block des Signalflußbilds angedeutet, können die Grund- und Bilanzgleichungen nur so weit zusammengefaßt werden, daß ω_M und M_R noch explizit enthalten sind. Genügt die Berücksichtigung der geschwindigkeitsproportionalen Reibung ($M_R = k\omega_M$), so können wiederum Übertragungsfunktionen errechnet werden.

Wird Bild 3.19 in Bild 3.20 umgezeichnet, dann sind innere und äußere Schleife sofort erkennbar. Die innere Schleife ist mittels (3.17) in die Zwischenfunktion $G_{H4}(p)$ überzuführen

$$G_{H4}(p) = \frac{\dfrac{1}{m_{red}p}}{1+\dfrac{1}{m_{red}p}\,(i_1 i_2)^2 k} = \frac{1}{(i_1 i_2)^2 k + m_{red}p}\,.$$

Danach sind die Ansätze für die gesuchten Übertragungsfunktionen denen zu (3.47) und (3.48) ähnlich. Es muß lediglich die Funktion $1/(m_{\mathrm{red}}p)$ durch die Zwischenfunktion $G_{\mathrm{H4}}(p)$ ersetzt werden.

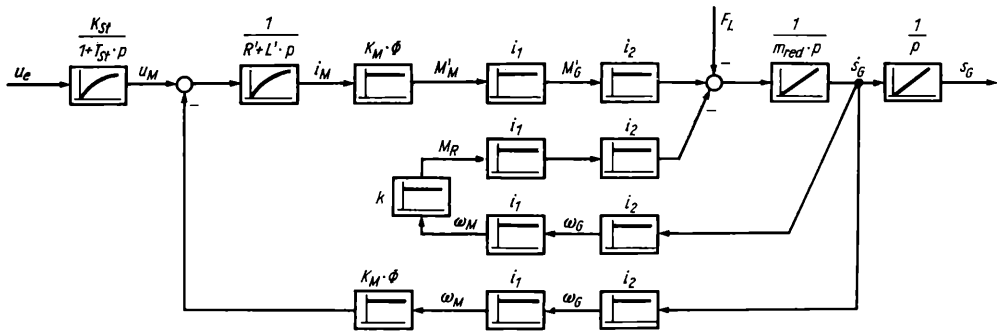


Bild 3.20. Linearisiertes und umgeformtes Signalflußbild des elektrischen Antriebs

Es ergeben sich infolge $k \neq 0$ nach den entsprechenden Umrechnungen kompliziertere Ausdrücke. Vor allem geht die eindeutige Zuordnung der Parameter zur Anker- und zur elektromechanischen Zeitkonstante verloren. Die Übertragungsfunktionen lauten jetzt

$$\frac{s_{\mathrm{G}}(p)}{u_{\mathrm{e}}(p)} = \frac{K_u}{p}\,\frac{1}{1 + T_1 p + T_2^2 p^2}\,\frac{1}{1 + T_{\mathrm{St}} p} \tag{3.51}$$

$$\frac{s_{\mathrm{G}}(p)}{F_{\mathrm{L}}(p)} = \frac{-K_F}{p}\,\frac{1 + T'_{\mathrm{A}} p}{1 + T_1 p + T_2^2 p^2} \tag{3.52}$$

mit

$$K_u = \frac{K_{\mathrm{St}} K_{\mathrm{M}} \Phi i_1 i_2}{i_1^2 i_2^2 (K_{\mathrm{M}} \Phi^2 + R'k)}$$

$$K_{\mathrm{F}} = \frac{R'}{i_1^2 i_2^2 (K_{\mathrm{M}}^2 \Phi^2 + R'k)}$$

$$T_1 = \frac{R' m_{\mathrm{red}} + L' k i_1^2 i_2^2}{i_1^2 i_2^2 (K_{\mathrm{M}}^2 \Phi^2 + R'k)}$$

$$T_2^2 = \frac{L' m_{\mathrm{red}}}{i_1^2 i_2 (K_{\mathrm{M}}^2 \Phi^2 + R'k)}$$

sowie T_{St} gemäß (3.42) und T'_{A} gemäß (3.51).

Daß die Reibung bei vorhandenem Getriebe eine große Rolle spielen kann, zeigt der Term $L' k i_1^2 i_2^2$ in der Zeitkonstante T_1, die die Dämpfung des Systems beeinflußt [siehe Gl. (3.18): $T_1 \mathrel{\hat{=}} 2\,DT$].

Es sei noch darauf hingewiesen, daß die Übertragungsfunktionen auch gelten, wenn nur eines der beiden Getriebe eingesetzt werden muß. Bei fehlendem Übersetzungsgetriebe wird i_1 zu $i_1 = 1$. Wird nur das Übersetzungsgetriebe verwendet, so fällt aus diesen Gleichungen i_2 heraus, und es werden der Weg s_{G} zum Winkel φ_{G}, die Belastungskraft F_{L} zum Belastungsmoment M_{L} und die reduzierte Masse m_{red} zum reduzierten Massenträgheitsmoment J_{red}.

3.3.3. In Lageregelungen eingesetzte pneumatische Antriebe

Praktische Bedeutung haben zwei Konzeptionen erlangt:

- der elektropneumatische Antrieb und
- der pneumatische Antrieb mit elektrischer Bremse.

Elektropneumatischer Antrieb

Es besteht eine sehr große Analogie zum ausführlich beschriebenen elektrohydraulischen Antrieb: ein dem Servoventil ähnliches Ventil, elektrisch ansteuerbar, dosiert den Luftstrom zu einem rotatorischen Verbraucher. Zwei wesentliche Unterschiede zwischen Pneumatik und Hydraulik erfordern jedoch für diesen Antrieb zusätzliche Maßnahmen:

1. Luft ist viel stärker kompressibel als Öl.
2. Vor allem wegen dieser Kompressibilität sind aus Sicherheitsgründen nur Druckquellendrücke bis zu 1 MPa möglich.

Um trotzdem eine ausreichend kleine Eigenzeitkonstante T [s. Gl. (3.21)] bei gleichzeitig ausreichend großem Stellmoment zu erzeugen, sind ein Getriebe mit sehr hoher Übersetzung und eine extrem kleine Ventil-Verbraucher-Einheit nötig.

Ein günstiger Verbraucher ist in diesem Fall eine Turbine mit sehr kleinem Verdrängungsvolumen V_g, an deren Gehäuse das Steuerventil angeflanscht ist. Als Getriebe kommen meist Zahnriementriebe in mehreren Stufen zum Einsatz. Damit ist ein Übersetzungsverhältnis bis etwa 10^3 zu erreichen. Im Bild 3.21 a ist das Schema eines solchen Antriebs dargestellt, im Bild 3.21 b ein aus den Bildern 3.12a und 3.19 (Getriebe) abgeleitetes Signalflußbild.

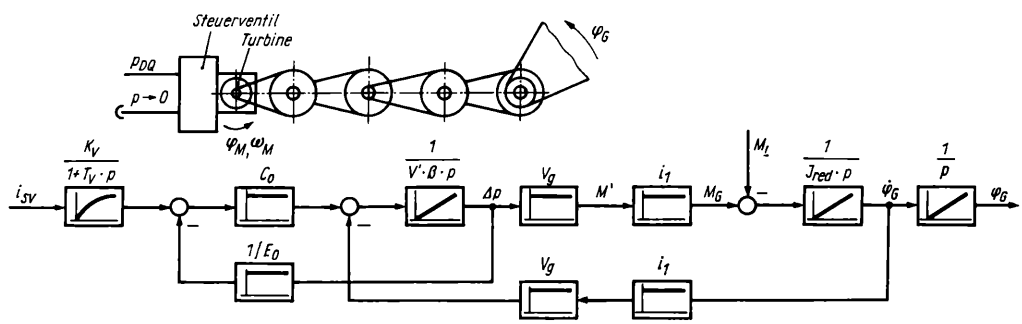


Bild 3.21. Elektropneumatischer Antrieb
a) Schema; b) linearisiertes und vereinfachtes Signalflußbild

Der Einfachheit halber wurden für den folgenden qualitativen Vergleich der Leckleitwert (G_L) und die geschwindigkeitsproportionale Reibung (k) vernachlässigt.

Wird dieses Signalflußbild mit dem im Bild 3.12a verglichen, so ist zu erkennen, daß der elektropneumatische Antrieb ähnliches dynamisches Verhalten wie der elektrohydraulische erreicht

- wenn das Produkt aus $V_g i_1$ viel größer als das geometrische Verdrängungsvolumen V_g eines hydraulischen Verbrauchers ist, um die wesentlich kleinere Druckverstärkung E_0 auszugleichen, und
- wenn die wesentlich größere Kompressibilität von Luft durch ausreichend kleines Volumen V' kompensiert wird.

Pneumatischer Antrieb mit elektrischer Bremse

Diese Speziallösung [3.6] soll nicht ausführlich mathematisch beschrieben werden, da ihr Modell erheblich von anderen abweicht. Der Pneumatikzylinder wird von einem einfachen Zwei- oder Dreipunktstellglied angesteuert und gibt die Bewegungsrichtung des Antriebs vor. Diese Aufgabe hat im elektrischen Antrieb die Umkehrschaltung des Stellglieds. Eine Bremse erzeugt eine Kraft, die der Bewegung immer entgegengesetzt ist. Die Ansteuerschaltung der Bremse realisiert die Aufgabe des Stromrichters im Stellglied des elektrischen Antriebs. Die vom Antrieb erzeugte Gesamtstellkraft ist im stationären Betrieb immer die Differenz der Druck- und der Bremskraft.

Es müssen beide Stellglieder im Signalfluß des Lageregelkreises liegen. Das dynamische Verhalten in der Nähe der Regelabweichung Null ist schwer beschreib- und beherrschbar.

3.3.4. Elektrische Schrittantriebe

Diese Antriebe nehmen gegenüber allen bisher beschriebenen eine Sonderstellung ein. In ihnen werden diskrete Eingangssignale in diskrete Weg- oder Winkelelemente umgesetzt. Bei Beachtung der Grenzen des dynamischen Verhaltens sind Positionen bei reiner Steuerung (keine Wegmessung, keine Anschläge, Endschalter usw.) exakt und freiprogrammierbar zu realisieren.

Sie werden mit elektrischen Impulsen angesteuert. Der Drehwinkel ist proportional der Pulszahl Z:

$$\varphi = \Delta\varphi Z; \qquad \Delta\varphi \text{ Schrittwinkel.}$$

Das Wirkprinzip besteht darin, daß eine Vorzugsrichtung des magnetischen Feldes des Ständers aufgebaut wird, die schrittweise wandert. Die technischen Realisierungen sind vielgestaltig (Mehrphasenstator oder Mehrstatorenausführungen in Verbindung mit unterschiedlichen Rotorprinzipien). Ihr Einsatz in der Industrierobotertechnik ist nur begrenzt möglich, da leicht Schrittfehler auftreten können, die infolge fehlender Lagemessung nicht wieder kompensiert werden.

Schrittverluste treten auf, wenn Schrittfrequenz und auf den Motor bezogenes Gegenmoment infolge äußerer Kräfte oder auch Massenwirkungen nicht exakt aufeinander abgestimmt sind.

Eine Modellierung im bisherigen Sinne ist nicht möglich und nicht notwendig. In der Bewegungssteuerung müssen die zulässigen Pulszahlen je Zeiteinheit berücksichtigt werden. Sie sind in der Start- und in der Bremsphase niedriger als bei stationärer Geschwindigkeit.

3.3.5. Einfachantriebe

Als Einfachantriebe sollen Antriebe gelten, die nicht in Lageregelungen eingesetzt werden und ein relativ einfaches Stellglied besitzen. Es kommen vor allem wegeventilgesteuerte pneumatische und hydraulische Antriebe zum Einsatz. Elektrische Einfachantriebe sind in der Handhabetechnik aus technisch-ökonomischen Gründen selten.

Die Wegeventile haben Zwei- oder Dreipunktverhalten zur Realisierung von Rechtslauf, Linkslauf und ggf. Halt. Letzteres ist dann nötig, wenn nicht gegen Anschläge gefahren, sondern z. B. mit Hilfe von Endschaltern auf Stillstand geschaltet wird. Ist Vorabschaltung auf niedrige Geschwindigkeit notwendig, sind zwei weitere Schaltzustände

nötig. Das wird durch zusätzliche Wege- und Drosselventile realisiert. Die Modellierung dieser Antriebe im Sinne einer optimalen Auslegung der Bewegungssteuerung reduziert sich im wesentlichen auf die Ermittlung von Verfahrzeiten zwischen den Positionen. Diese werden überwiegend experimentell ermittelt.

3.4. Meßsysteme

3.4.1. Aufgaben

Die automatische Steuerung eines technischen Systems erfordert die Erfassung von momentanen Zustandsparametern, wenn diese aufgrund unzureichender A-priori-Kenntnisse oder aus Aufwandsgründen nicht in dem der Steuerung zugrunde liegenden Modell mit ausreichender Genauigkeit enthalten sind. Dies geschieht mittels geeigneter Meßsysteme oder Sensoren.

In der klassischen Meßtechnik ist das Ergebnis der Messung ein Zahlenwert, der die Meßgröße als Vielfaches einer geeignet gewählten Einheit charakterisiert. Die gesuchte, i.allg. nichtelektrische Größe wird durch Ausnutzung unterschiedlicher physikalischer Effekte in ein elektrisches Signal gewandelt, zweckentsprechend vorverarbeitet und ausgegeben (Bild 3.22). Die Vorverarbeitung dient der Eliminierung von Störungen, Kompensation unerwünschter Wandlereigenschaften und Verstärkung des elektrischen Abbildsignals. Zur Übertragung und Anpassung an das Steuerungssystem können weitere Signalwandlungen vorgenommen werden. Diese Form der Informationsgewinnung ist nur geeignet, wenn der gesuchte Prozeßparameter der direkten Messung zugänglich, d.h. über einen geeigneten Meßwandler erfaßbar ist. Mit zunehmender Komplexität der Steuerungsaufgabe werden Informationen über Prozeßzustände benötigt, die nicht direkt gemessen werden können und in der Regel aus einer Vielzahl von Einzelmessungen abgeleitet werden. Eine für die Robotertechnik typische Aufgabe ist die Erkennung eines im Arbeitsraum befindlichen Werkstücks. Die Steuerungsaufgabe besteht darin, N unterschiedliche Teile an ebenso vielen unterschiedlichen Ablageplätzen zu stapeln.

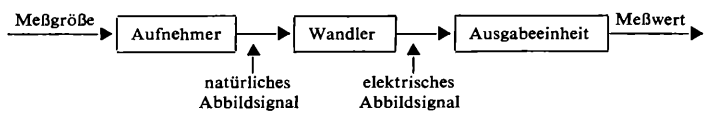


Bild 3.22. Grundschema einer Meßkette

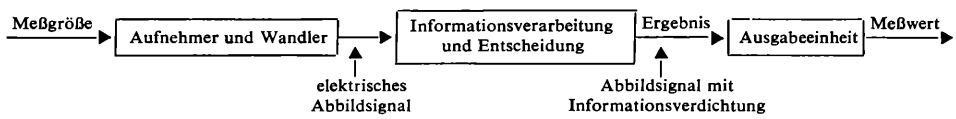


Bild 3.23. Schema eines Sensorsystems

Der Ausgang des Meß- oder Sensorsystems muß also je nach Art des Teiles N unterschiedliche Ausgangssignale liefern. Kommt als Wandler eine Kamera zum Einsatz, so liefert diese ein zweidimensionales elektrisches Abbild des Werkstücks. Zur Erzeugung der N Ausgangssignale muß daher die Meßkette in der im Bild 3.23 gezeigten Weise erweitert werden. Anhand von Modellabbildungen der N Werkstücke wird durch einen Vergleich die Übereinstimmung mit dem erfaßten Bild ermittelt und bei minimaler Abweichung

eine Entscheidung über die Zugehörigkeit des Werkstücks zu einer der N Klassen getroffen.

Für die Steuerung von Robotern benötigte Informationen sind die Zustandsparameter des mechanischen Systems und Parameter aus der technologischen Umwelt, mit der der Roboter in Wechselwirkung tritt.

Sensoren zur Bestimmung des momentanen Roboterzustands (Position und Geschwindigkeit der Achsen, Kräfte in den Gelenken usw.) werden häufig als „innere Sensoren", Sensoren zur Erfassung des Umweltzustands als „äußere Sensoren" bezeichnet. Ein Grenzfall tritt auf, wenn beispielsweise durch Kraftsensoren im Roboterarm sowohl Schwerkräfte und dynamische Kräfte des mechanischen Systems als auch durch den technologischen Prozeß bedingte Wechselwirkungskräfte mit der Umwelt erfaßt werden.

Äußere Sensoren werden im Abschnitt 7. ausführlich beschrieben.

Allgemeine Probleme zur Auslegung von Meßsystemen und Eigenschaften von Wandlern sowie die Wirkprinzipien der am häufigsten eingesetzten inneren Sensoren werden nachfolgend erläutert. Auf nicht roboterspezifische Sensoren zur Kontrolle des inneren Betriebszustands, wie Druck, Temperatur, Verschleiß usw., wird an dieser Stelle nicht eingegangen.

Die Begriffe Meßwandler, Meßsystem, Sensor, Sensorsystem werden in der Fachliteratur nicht einheitlich benutzt, da durch zunehmende räumliche und funktionelle Integration der Verarbeitungsstufen gemäß den Bildern 3.22 und 3.23 die Grenzen häufig nicht mehr erkennbar sind.

Nachfolgend werden deshalb neben dem Begriff des eigentlichen Meßwandlers die Begriffe Sensor, Meß- oder Sensorsystem für eine Einrichtung benutzt, die die zur Steuerung benötigten Ausgangssignale entsprechend den Bildern 3.22 und 3.23 liefert.

3.4.2. Probleme der Meßsignalgewinnung

Besondere Anforderungen an die Meßtechnik in Verbindung mit der Robotersteuerung resultieren aus der unmittelbaren Kopplung an den technologischen Prozeß und der damit verbundenen Zielstellung, Erzeugnisse gewünschter Menge und Qualität zu produzieren. Das erfordert die Beachtung solcher Parameter wie

- Zuverlässigkeit bei Einsatz unter rauhen Umgebungsbedingungen
- Meßbereich, Meßgenauigkeit und Langzeitkonstanz
- Reproduzierbarkeit der Meßwerte
- Störsicherheit, fehlerfreie Meßwertübertragung
- Dynamik, sofortige Verarbeitung der Information.

Meßgrößenwandlung zur analogen und diskreten Informationsgewinnung

Die in den meisten Fällen nichtelektrische, analoge Meßgröße ist zum Zweck der Übertragung sowie der Weiterverarbeitung im Prozessor auf ein von diesem verarbeitbares, digitales elektrisches Signal abzubilden. Die Wandlung des Signals innerhalb der im Bild 3.22 dargestellten Meßkette erfolgt i. allg. in mehreren Schritten, wobei die Meßgröße zunächst auf einen analogen elektrischen Informationsparameter abgebildet wird, z. B. Amplitude, Frequenz oder Phasenlage einer Spannung.

Je nach dem Wertevorrat am Ausgang der Meßkette unterscheidet man *analoge* und *diskrete* Signale. Im folgenden wird an Beispielen der Signalfluß in der Meßkette demonstriert. Die sich daraus ergebenden Konsequenzen für das dynamische Verhalten werden anschließend betrachtet.

Bild 3.24 zeigt den Signalfluß in einem Kraftmeßumformer mit analoger Informationsgewinnung. Die Kraftänderung wird im Meßfühler in eine Wegänderung (Formänderung eines deformierbaren Körpers) umgewandelt. Mittels Dehnungsmeßstreifen, die auf dem elastischen Körper befestigt sind, wird eine der Wegänderung proportionale Widerstandsänderung realisiert. Die Dehnungsmeßstreifen sind in einer Brückenschaltung angeordnet. Mit einer Trägerfrequenzmeßeinrichtung wird die Widerstandsänderung auf eine zugeordnete Spannungsänderung abgebildet, die als analoges elektrisches Ausgangssignal den Meßwert darstellt. Die Umsetzung in ein rechnerkompatibles digitales Signal erfolgt in einem nachzuschaltenden Analog-Digital-Umsetzer.

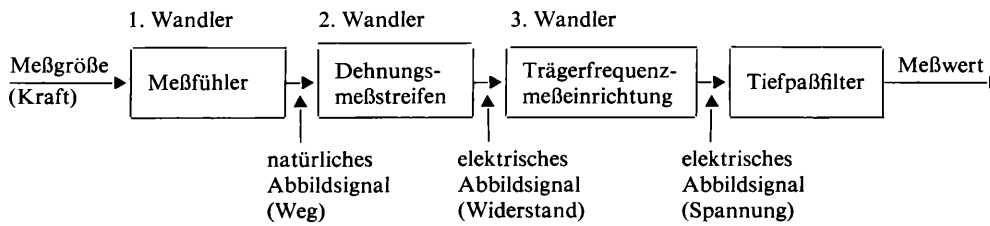


Bild 3.24. Signalfluß in einer Meßkette mit analoger Meßwertausgabe

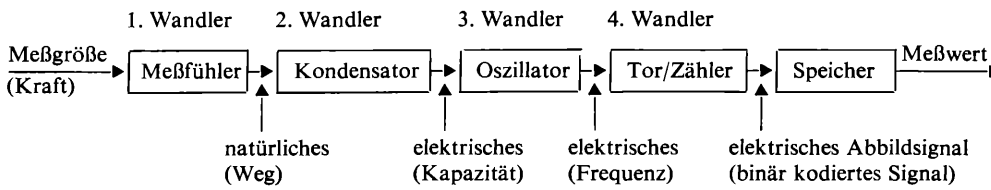


Bild 3.25. Meßkette für die Messung einer Kraft mit digitaler Meßwertausgabe

Eine Meßkette für diskrete Informationsgewinnung, ebenfalls für die Kraft als Eingangsgröße, wird im Bild 3.25 gezeigt. Die Kraftänderung bewirkt die Lageänderung der Platte eines Kondensators, der frequenzbestimmendes Element eines elektrischen Schwingkreises ist. In diesem Fall ist die Oszillatorfrequenz ein analoges Abbildsignal der Kraft. Frequenzanaloge Meßverfahren sind besonders „digitalfreundlich". Der letzte Wandler in der Meßkette setzt die Frequenz unmittelbar in ein digitales Ausgangssignal um. Dazu wird während einer definierten Meßzeit die Zahl der Perioden des Abbildsignals in einen Binärzähler eingezählt. Am Ausgang des Zählers steht nach Ablauf der Meßzeit der kodierte, digitale Meßwert an und kann bis zur Bereitstellung des folgenden Meßwerts in einen Speicher übernommen werden.

Eine andere Möglichkeit zur Digitalisierung besteht darin, während einer Periode des frequenzanalogen Signals (oder während einer konstanten Anzahl von Perioden) eine Torschaltung zu öffnen und während dieser Zeit Taktimpulse konstanter Folgefrequenz in einem Zähler zu addieren.

Der Umsetzfehler F ergibt sich in beiden betrachteten Varianten zu

$$F = (fT)^{-1}. \tag{3.53}$$

Dabei kennzeichnet f die zu bestimmende Frequenz und T die definierte Meßzeit bzw. f die Folgefrequenz der Taktimpulse und T die Periodendauer des frequenzanalogen Signals (Toröffnungszeit). Soll der Meßfehler $1\,^0/_{00}$ nicht übersteigen, so sind mindestens 10^3 Impulse je Messung einzuzählen. Entsprechend groß muß die definierte Meßzeit gewählt werden bzw. entsprechend hoch die Folgefrequenz der Taktimpulse. Die tech-

nische Realisierbarkeit wird die Wahl der einen oder anderen Methode bestimmen [3.11]. An analoge Baugruppen einer Meßkette mit digitaler Meßwertausgabe werden hohe Anforderungen hinsichtlich Meßbereichsdynamik und Konstanz gestellt, wenn die hohe erreichbare Auflösung digitaler Verfahren genutzt werden soll. Deshalb sind Meßketten anzustreben, in denen die Signalverarbeitung ausschließlich digital erfolgt. Dazu muß bereits im Meßfühler die Meßgröße auf ein diskretes, ggf. auch kodiertes elektrisches Signal abgebildet werden, oder die Meßgröße selbst liegt bereits quantisiert vor. Solche Meßketten lassen sich in relativ wenigen Fällen realisieren. Ein Beispiel ist der kodierte Längenmaßstab.

Dynamische Eigenschaften von Wandlern

Das dynamische Übertragungsverhalten der Funktionsgruppen einer Meßkette hat wesentlichen Einfluß auf die Realisierbarkeit einer zeitoptimalen Robotersteuerung. Der Zeitverlauf der Meßgröße soll auch bei schnellen Bewegungsabläufen am Ausgang der Meßkette möglichst unverfälscht auf den elektrischen Signalparameter abgebildet werden.

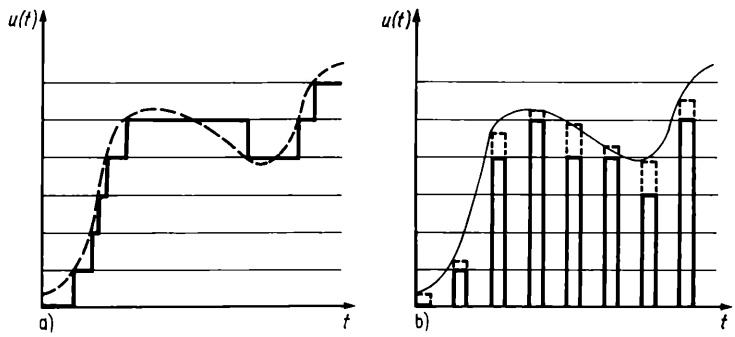


Bild 3.26
Signalarten, untergliedert nach dem Zeitverlauf und dem Wertevorrat
a) kontinuierlich
b) diskontinuierlich
– – – analog;
——— diskret

Nach dem Zeitverlauf des Informationsparameters am Meßkettenausgang unterscheidet man *zeitkontinuierliche* und *zeitdiskrete* Signale (Bild 3.26). Dynamische Eigenschaften von Meßsystemen werden damit jedoch nicht hinreichend charakterisiert. Dazu die Betrachtung folgender Beispiele:

Die im Bild 3.24 skizzierte Meßkette liefert ein analoges, zeitkontinuierliches Ausgangssignal. Bei einer Trägerfrequenzbrückenspannung von 5 kHz wird die Grenzfrequenz für dieses Meßsystem, bedingt durch das erforderliche Tiefpaßglied zum Ausfiltern des 5-kHz-Trägers, etwa 1 kHz betragen (Bild 3.27). Daraus resultiert für das Meßsystem eine Eigenzeitkonstante T in der Größenordnung von 1 ms. Bei einer sprunghaften Wegänderung ist prinzipiell erst nach Ablauf einer vollen Periode T_f der Trägerfrequenz-

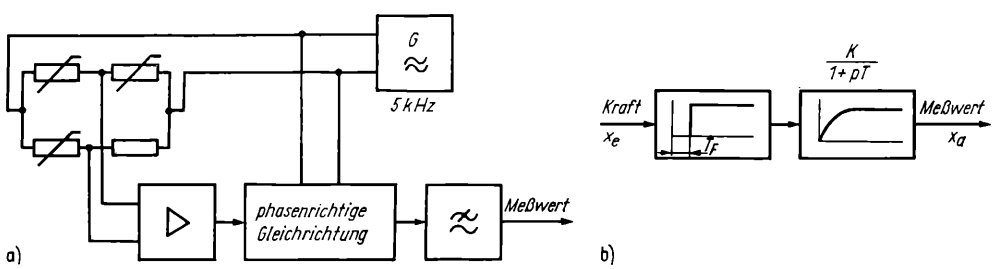


Bild 3.27. Meßschaltung für eine Widerstandsbrücke mit Dehnungsmeßstreifen
a) Blockschaltbild; b) Übertragungsverhalten
K statischer Übertragungsfaktor

spannung eine Amplitudenbestimmung ausführbar. Das regelungstechnische Übertragungsverhalten wird demnach näherungsweise durch ein Laufzeitglied mit Verzögerung bestimmt (Bild 3.27b). Hinzu kommt schließlich eine weitere Verzögerung, die von der Umsetzzeit des nachzuschaltenden Analog-Digital-Umsetzers bestimmt wird.

Die im Bild 3.25 dargestellte Meßkette liefert ein digitales, zeitdiskretes Ausgangssignal. Die dynamischen Eigenschaften dieser Meßanordnung werden von der technisch realisierbaren Möglichkeit der Umsetzung der Oszillatorfrequenz in ein digitales, kodiertes Ausgangssignal bei Einhaltung des zulässigen Fehlers bestimmt. Die Umsetzung der Frequenz f in eine digitale Ausgangsgröße kann über die Bestimmung der Periodendauer T_f durch Auszählen der Zahl der Taktimpulse einer konstanten Impulsfolgefrequenz f_0 während einer Periode erfolgen (Bild 3.28). Bei einem zulässigen Fehler von $1^0/_{00}$ muß für die Folgefrequenz der Taktimpulse $f_0 \geqq 10^3 f$ erfüllt sein. Liegt die Frequenz f in der Größenordnung 1 MHz, so könnte zwar in Zeitabständen von 2 µs jeweils ein neuer Meßwert bereitgestellt werden, jedoch ist die technische Realisierbarkeit eines einfachen Meßsystems bei einer Taktfolgefrequenz von 1 GHz problematisch.

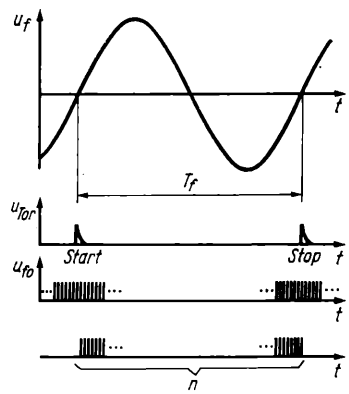


Bild 3.28
Prinzip der digitalen Periodendauermessung
n Impulszahl

Das Auszählen der Periodenzahl während einer definierten Zeit erfordert beim vorgegebenen Fehler eine Meßzeit von 1 ms. Nach Ablauf dieses Zeitintervalls kann jeweils ein neuer Meßwert am Ausgang der Meßkette abgefragt werden.

Schwingt dagegen der Oszillator mit einer Frequenz in der Größenordnung von 1 kHz, so ist die im Bild 3.28 angegebene Variante mit einer Taktfolgefrequenz von $f_0 = 1$ MHz realisierbar. Da für die Bildung eines Meßwerts mindestens eine Periode der Oszillatorfrequenz erforderlich ist, ergibt sich für den zeitlichen Abstand zwischen zwei Messungen wiederum $T_A \approx 1$ ms. Bei schnellen Bewegungsabläufen wird über Veränderungen im System während der Zeit T_A keine Information an die Steuerung gegeben. Das Übertragungsverhalten der Meßkette kann näherungsweise durch ein Laufzeitglied mit der Laufzeit $T_L = T_A/2$ beschrieben werden.

3.4.3. Weg- und Winkelmessung

3.4.3.1. Allgemeine Probleme

Positioniermeßsysteme haben bei der Programmierung des Roboters die aktuellen digitalen Meßsignale, die dem jeweiligen Ort zugeordnet sind, für die Abspeicherung als Sollwerte bereitzustellen. Bei der Programmwiedergabe verarbeitet die Steuerung die Meßwertsignale, um aus dem Sollwert-Istwert-Vergleich Steuerbefehle abzuleiten.

Bei ausschließlicher Teach-in-Programmierung sind erreichbare Auflösung und Reproduzierbarkeit der Meßwerte entscheidend für die Positioniergenauigkeit. Ist hingegen auch textuelle Programmierung vorgesehen, dann sind zusätzlich an die Genauigkeit bzw. Linearität des Meßsystems gleich hohe Anforderungen wie an die Auflösung zu stellen. Im diskreten Bereich ist eine Linearisierung hardwaremäßig mittels programmierbarer Speicher möglich. Die gemessenen Werte werden dann vom Ausgang der Meßkette nicht unmittelbar der Steuerung zugeführt, sondern sie stellen die Adressen für einen Speicherbaustein dar, in dem die zugeordneten linearisierten Meßwerte abgelegt sind. Diese Speicher stellen die Ausgabeeinheit der Meßkette dar. Der Speicheraufwand kann, abhängig vom Meßbereich und der erforderlichen Auflösung, sehr hoch sein; die Linearisierung erfolgt aber praktisch ohne zeitliche Verzögerung.

Eine softwaremäßige Linearisierung mittels Mikrorechner auf der Grundlage der analytischen Beschreibung der Wandlerkennlinie ist zwar bei wesentlich geringerem Speicherbedarf realisierbar, jedoch ist diese Lösung für eine Robotersteuerung zu zeitaufwendig, um mit verfügbaren Mikrorechnern bei schnellen Bewegungsabläufen im Echtzeitbetrieb arbeiten zu können.

In Abhängigkeit von der Art der Ankopplung der Meßwertaufnehmer an die Bewegungseinheiten des Roboters ist zwischen direkter und indirekter Positionsbestimmung zu unterscheiden. Eine translatorische Bewegung wird direkt erfaßt, wenn der bewegliche Teil eines translatorischen Meßwertaufnehmers starr mit der zu positionierenden Bewegungsgruppe verbunden ist. Wird hingegen die Lage der zu bewegenden mechanischen Baugruppe über bewegliche Elemente auf ein translatorisches Meßsystem oder über Zahnstange–Ritzel, Zahnriemen, Umlenkrolle oder Spindel auf einen rotatorischen Aufnehmer übertragen, so liegt eine indirekte Messung vor. Spiele in den Verbindungsgliedern sowie Deformationen zwischen der zu positionierenden Baugruppe und dem Ort der Meßwertaufnahme führen zu unkontrollierbaren Meßfehlern und verschlechtern die Reproduzierbarkeit der Positionierung. Dennoch überwiegt in der Praxis die indirekte Weg- oder Winkelmessung aufgrund der in der Regel einfacheren konstruktiven Anbaubedingungen, insbesondere für rotatorische Meßwertaufnehmer.

Nach dem Meßverfahren ist zwischen *absoluten* und *inkrementalen* Verfahren zu unterscheiden. Bei absolut arbeitenden Verfahren wird der Istwert eindeutig auf den Signalparameter am Ausgang der Meßkette abgebildet. Der Istwert ist dabei auf einen festen Referenzpunkt der jeweiligen Achse bezogen.

Äußere Störeinflüsse oder Ausfall der Stromversorgung bewirken keine bleibende Beeinflussung des Meßwerts. Kurzzeitige Fehler, beispielsweise durch Einkopplung von Störsignalen verursacht, werden nicht verschleppt.

Bei den inkrementalen Verfahren wird der Istwert durch Aufzählen von Weg- oder Winkelinkrementen von einem Anfangswert aus gebildet. Die Inkremente werden von der Meßverkörperung im Meßwertaufnehmer vorgegeben. Der Anfangswert ist bei jeder Inbetriebnahme der Meßeinrichtung durch Anfahren einer definierten Anfangsposition neu festzulegen (Nullung). Bei Ausfall der Stromversorgung geht die Information verloren. Äußere Störeinflüsse können zu Fehlerinkrementen im Vorwärts-Rückwärts-Zähler führen, die erst bei einer erneuten Nullung korrigiert werden. Um dem Verschleppen von Fehlerinkrementen entgegenzuwirken, bringen Hersteller inkremental arbeitender Meßwertaufnehmer mitunter in bestimmten Weg- bzw. Winkelabständen Referenzmarken an, die eine Korrektur von Fehlern beim Überfahren dieser Marken während der Messung ermöglichen (Bild 3.29).

Obwohl die absoluten Verfahren dem Erfordernis der Verarbeitung der absoluten Information in der Steuerung entgegenkommen, werden inkrementale Verfahren in der

Praxis häufiger angewendet. Die Gründe hierfür sind im einfacheren Aufbau, der geringeren Masse, des niedrigeren Preises inkrementaler Aufnehmer zu sehen sowie darin, daß der Meßbereich vom Meßgrößenaufnehmer nicht prinzipiell begrenzt wird und daß mit modernen elektronischen Auswerteverfahren eine weitgehende Störimpulsunterdrückung und damit eine hohe Zuverlässigkeit gewährleistet werden kann.

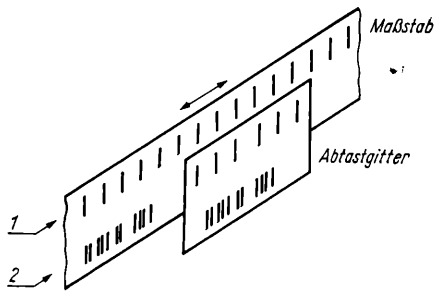


Bild 3.29
Inkrementaler Längenmaßstab mit Referenzmarke
1 Inkrementteilung; *2* Referenzmarke

Eine Zwischenstellung nehmen die zyklisch-absoluten Meßverfahren ein. Weit verbreitet sind phasenzyklische Systeme. Hinsichtlich der Gesamtmeßeinrichtung handelt es sich um inkremental arbeitende Wandler mit relativ großen Weg- bzw. Winkelschritten, z.B. 0,5 bis 10 mm bei der Wegmessung, 1 bis 360° bei der Winkelmessung. Innerhalb einer Schrittweite wird die Meßgröße analog interpoliert und absolut auf den elektrischen Signalparameter abgebildet. Zur exakten Nullung des Systems braucht eine Referenzposition nur grob – innerhalb der vorgegebenen Schrittweite – angefahren zu werden und nicht, wie bei rein inkremental arbeitenden Systemen, exakt auf die Referenzposition.

Funktionsweise und Eigenschaften einiger ausgewählter Wandler sollen dargestellt werden; auf dynamische Eigenschaften, die sich auf die Steuerung auswirken, wird hingewiesen.

3.4.3.2. Absolute Meßverfahren

Binäre Positionsgeber

Mit binären Positionsgebern sind meßtechnische Aufgaben mit geringem Schwierigkeitsgrad lösbar. Binäre Positionsgeber stellen keine Meßsysteme im eigentlichen Sinne dar. Sie liefern keine ständige Information über die Position, sondern signalisieren lediglich das Erreichen einer bestimmten Position, die Anwesenheit oder Nichtanwesenheit eines Werkstücks an einem definierten Ort. Die bereitgestellten Meßwerte können nicht die Basis einer kontinuierlichen Steuerung des Prozeßablaufs sein. Verschiedene Ausführungsformen sind bekannt, z.B. elektromechanische Schalter, induktive und kapazitive Schlitz- und Näherungsinitiatoren, Lichtschranken mit sichtbarem Licht oder Infrarotlicht bzw. auf der Basis der Kernstrahlung, Ultraschallinitiatoren.

Potentiometer

Präzisionspotentiometer werden als Weg- und Winkelgeber verwendet. Zwischen der Meßgröße s (Weg bzw. Winkel) und der Spannung U_s, die am Potentiometerschleifer abgegriffen wird, besteht ein linearer Zusammenhang (Bild 3.30). Der Meßfehler liegt in der Größenordnung 10^{-3} des Meßbereichs. Er wird durch die endliche Schleiferbreite, die Inhomogenität des Kunststoffschicht-Widerstandselements bzw. durch den Windungssprung bei Drahtpotentiometern bedingt. Potentiometer werden für Meßbereiche

bis 900 mm bei translatorischem und bis zu 20 Umdrehungen bei rotatorischem Aufbau angeboten. Der Meßwert ist eine analoge Spannung; ein Analog-Digital-Umsetzer muß nachgeschaltet werden. Letzterer bestimmt die erreichbare Meßwertfolge, die im ms- bis µs-Bereich liegen kann.

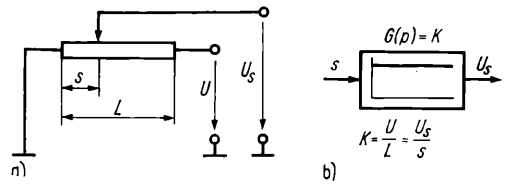


Bild 3.30
Potentiometer als Weggeber
a) Wirkprinzip; b) Übertragungsverhalten
K Übertragungsfaktor

Potentiometer sind einfach im Aufbau und ermöglichen bei Verwendung schneller Analog-Digital-Umsetzer eine hohe dynamische Auflösung. Die nicht berührungslose, mit Materialverschleiß verbundene Meßwertabnahme ist jedoch ein entscheidender Nachteil. Daraus resultieren zusätzliche Meßfehler durch sich zeitlich ändernde Übergangswiderstände und Störanfälligkeit bei hohen Beschleunigungen infolge Prellerscheinungen am Schleifer. Letzteres kann bei der Analog-Digital-Umsetzung große Fehler verursachen.

Kodierte Längen- und Winkelmaßstäbe

Kodelineale und Kodescheiben mit zumeist fotoelektrischem Meßwertabgriff sind Meßwertaufnehmer, die die Meßgröße unmittelbar und nahezu verzögerungsfrei auf ein digitales, kodiertes, elektrisches Signal abbilden. Einschrittige Kodes, wie der Gray-Kode, werden bevorzugt verwendet [3.12] (Bild 3.31). Die einzelnen Spuren dieses Kodes sind zwar nicht bewertet, so daß eine nachträgliche Kodeumsetzung notwendig ist. Mehrschrittige Kodes erfordern aber einen höheren Abtastaufwand, um die Eindeutigkeit des Meßwerts bei gleichzeitiger Änderung des Kodemusters in mehreren Spuren zu gewährleisten. Die dynamische Auflösung kann durch die dynamischen Parameter der Bauelemente für die optisch-elektrische Meßwertumsetzung begrenzt sein.

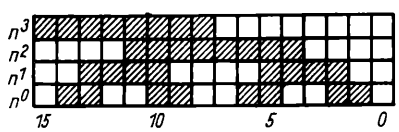


Bild 3.31
Kodierter Längenmaßstab mit einschrittigem Gray-Kode

Kodelineale und -scheiben entsprechen aus steuerungstechnischer Sicht allen Forderungen, die an Meßwertaufnehmer für Robotersteuerungen gestellt werden. Bei üblicherweise erforderlichen Meßbereichen und Auflösungen sind aber Kodemuster mit sehr vielen Spuren notwendig. Solche Umsetzer haben große Abmessungen und sind sehr teuer. Die Störanfälligkeit bei fotoelektrischem Meßwertabgriff und hoher Auflösung ist problematisch. Dies sind Gründe dafür, daß kodierte Längen- und Winkelmaßstäbe nicht in großem Umfang in der Handhabetechnik eingesetzt werden.

3.4.3.3. Inkrementale Meßverfahren

Beim inkrementalen Wegaufnehmer mit fotoelektrischem Meßwertabgriff ist auf einem Maßstabträger aus Stahl oder Glas über den gesamten Meßweg eine Gitterteilung konstanter Schrittweite in Form alternierend aufeinanderfolgender reflektierender und nichtreflektierender Marken aufgebracht. Ein in geringem Abstand darüber in Meßrichtung

verschiebbarer Abtastkopf enthält vier Abtastfelder mit Gitterteilungen der gleichen Periode und den fotoelektrischen Wandlern. Die vier Gitterteilungen sind jeweils um ein Viertel ihrer Periode gegeneinander versetzt angeordnet. Damit ist eine Eliminierung des störenden Gleichanteils des Ausgangssignals möglich, und eine Entscheidung über die Zählrichtung des nachgeschalteten Vorwärts-Rückwärts-Zählers kann getroffen werden (Bild 3.32). Am Zählerausgang steht stets der aktuelle kodierte Meßwert unverzögert an.

Die Auswertung kann in jedem der vier Abtastfelder des Abtastkopfs über eine große Gitterlinienzahl erfolgen; auch bei kleiner Gitterkonstante wird eine für die Messung hinreichende Lichtintensität erreicht. Im Gegensatz dazu kann beim kodierten Längenmaßstab in der am höchsten auflösenden Spur nur eine Gitterteilung ausgewertet werden. Daraus resultiert die höhere mögliche Auflösung inkrementaler Verfahren.

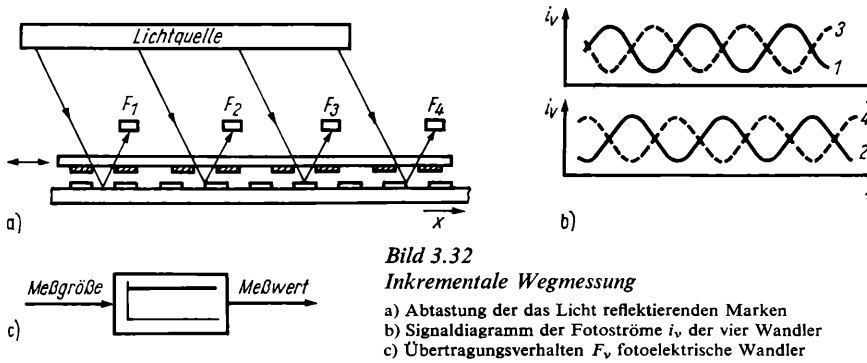


Bild 3.32
Inkrementale Wegmessung
a) Abtastung der das Licht reflektierenden Marken
b) Signaldiagramm der Fotoströme i_V der vier Wandler
c) Übertragungsverhalten F_V fotoelektrische Wandler

Der Erleichterung der fotoelektrischen Signalauswertung dient die Moiréstreifentechnik. Durch schräge Anordnung der Gitter im Abtastkopf relativ zum Maßstab wird eine scheinbare Vergrößerung der Gitterabstände erreicht (Bild 3.33). Größere Hell-Dunkel-Zonen können ausgewertet werden, und es wird über viele Gitterlinien gemittelt.

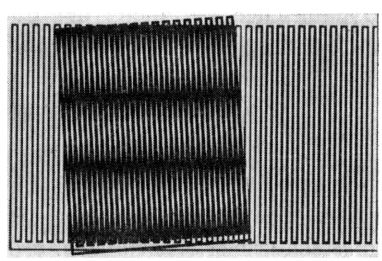


Bild 3.33
Moiréstreifentechnik zur scheinbaren Vergrößerung der Gitterabstände

Die maximale Verfahrgeschwindigkeit ist bei inkrementalen Verfahren begrenzt; sie wird durch die endliche Grenzfrequenz der eingesetzten fotoelektrischen Wandler bestimmt. Von besonderer Bedeutung ist unter diesem Gesichtspunkt die störsichere Funktionsweise des Richtungsdiskriminators, wenn der Wandler hochfrequenten mechanischen Erschütterungen ausgesetzt ist, da einmal eingezählte Fehlerimpulse bis zur nächsten Nullung als Meßfehler verschleppt werden.

Nach dem gleichen Prinzip arbeiten inkrementale Rundmeßsysteme. Maßstab und Abtastgitter sind dabei auf einer Kreisscheibe bzw. einem Kreissegment angeordnet. Solche Systeme werden mit 100 bis zu 36000 Linien/Umfang angeboten. Häufig wird eine elektronische Vervielfachung der binären Ausgangsimpulse zur Verbesserung der Auflösung angewendet. Auch Wandler mit elektromagnetischer Abtastung werden verwendet.

3.4.3.4. Phasenzyklische Meßverfahren

Resolver

Der zweipolige Resolver ist ein spezieller, schleifringloser Drehmelder. Beim Phasenansteuerverfahren werden in die zwei geometrisch um 90° versetzt angeordneten Statorwicklungen zwei harmonische, elektrisch um 90° phasenverschobene Ströme gleicher Amplitude eingespeist (Bild 3.34):

$$i_1(t) = I_0 \sin \omega t, \quad i_2(t) = I_0 \cos \omega t. \tag{3.54}$$

In der Rotorwicklung wird eine Spannung u_R induziert, die sich aus der Überlagerung der zwei induzierten Teilspannungen ergibt:

$$u_R(t) = U_0 \sin (\omega t + \alpha). \tag{3.55}$$

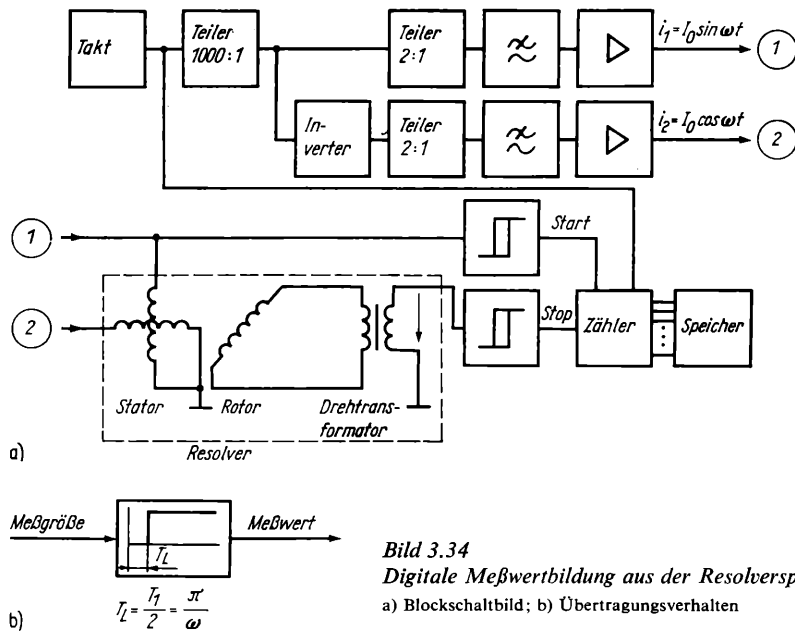


Bild 3.34
Digitale Meßwertbildung aus der Resolverspannung
a) Blockschaltbild; b) Übertragungsverhalten

Das ist eine Spannung der gleichen Winkelfrequenz ω. Die Amplitude U_0 ist bei geeigneter Polgestaltung konstant; der Phasenwinkel α ist abhängig vom Drehwinkel des Rotors relativ zum Stator. Er ändert sich bei einer vollen Rotorumdrehung zwischen 0° und 360° und stellt ein analoges Abbildsignal der Meßgröße dar. Die Umsetzung in einen digitalen kodierten Meßwert kann ebenso erfolgen, wie es im Bild 3.28 für ein Frequenzsignal skizziert wurde. Praktisch wird eine Auflösung der vollen Periode in 2000 Teilschritte realisiert. Beträgt die Frequenz der Resolverspannung 1 kHz, so muß die Taktimpulsfolgefrequenz f_0 dann 2 MHz betragen.

Der Meßwertaufnehmer arbeitet für eine Rotorumdrehung absolut, die Zahl der Umdrehungen einschließlich Vorwärts-Rückwärts-Entscheid wird inkremental erfaßt.

Die dynamischen Eigenschaften ergeben sich aus den Darlegungen im Abschnitt 3.4.2. (s. auch Bild 3.34b).

Zur Anpassung an die jeweiligen mechanischen Baugruppen des Roboters werden Resolver z.T. mit Meßgetrieben verwendet. Getriebe bewirken aber stets zusätzliche

Meßfehler. Der Einsatz mehrpoliger Resolver (bis 10polig) wird deshalb vorgezogen. Der Bereich, in dem die Meßgröße absolut auf den Signalparameter abgebildet wird, erstreckt sich beim 10poligen Resolver nur über ein Fünftel einer Umdrehung. Die Auflösung ist um einen Faktor 5 höher als beim zweipoligen Resolver.

Die Verfahrgeschwindigkeit ist begrenzt, da nur endlich viele Meßwerte je Zeiteinheit am Ausgang der Meßkette bereitgestellt werden. Diese müssen eine eindeutige Erfassung der inkremental zu zählenden Grobintervalle durch die Steuerung einschließlich Richtungsentscheid ermöglichen.

Inductosyn

Meßprinzip und Auswerteverfahren sind prinzipiell die gleichen wie beim Resolver. Beim Rundinductosyn sind Stator- und Rotorwicklungen jeweils in einer Ebene auf Kreisscheiben angeordnet, die sich im Abstand von wenigen Zehntelmillimetern gegenüber stehen (Bild 3.35). Der Grundkörper besteht aus nichtmagnetischem Material; die Frequenz der eingespeisten Ströme liegt im Bereich 2,5 bis 100 kHz. 180-, 360- und 720-polige Rotorwicklungen sind üblich. Bei einem 360poligen Rotor sind 180 Leiterzeugperioden symmetrisch auf dem Vollkreis angeordnet; eine Leiterzugperiode entspricht einer vollen Umdrehung beim zweipoligen Resolver. Eine Auflösung von 1 Winkelsekunde ist realisierbar.

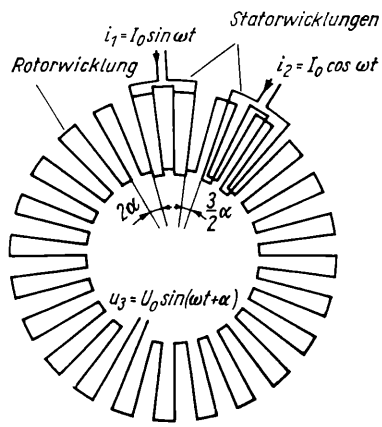


Bild 3.35. Rundinductosyn – Prinzipskizze der Stator- und Rotorwicklung

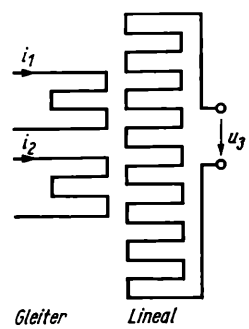


Bild 3.36. Linearinductosyn – Prinzipskizze der Lineal- und Gleiterwicklung

Bei Transformation der Leiterzüge des Rundinductosyns von Zylinderkoordinaten in kartesische Koordinaten erhält man aus der Wicklung des Stators die des Gleiters und aus der Wicklung des Rotors die des Lineals für das Linearinductosyn (Bild 3.36). Für Positionieraufgaben an Robotern ist wegen der geringen Masse das Bandinductosyn interessant, bei dem die mäanderförmigen Leiterzüge des Lineals auf einem etwa 0,2 mm dicken, nichtmagnetischen Stahlband aufgebracht sind, das sich in einer Führung des Gleiters bewegt. Die Leiterzugperiode beträgt im allgemeinen 2 mm, die Auflösung beim Bandinductosyn 10 μm. Induktive phasenzyklische Meßsysteme werden aufgrund ihrer Robustheit, ihrer Störsicherheit auch bei Verschmutzung sowie der hohen Auflösung und der einfachen Anpassung an die Steuerung häufig eingesetzt.

Auf weitere phasenzyklische Meßsysteme mit magnetischem, optoelektronischem und kapazitivem Wandler soll an dieser Stelle hingewiesen werden [3.13] [3.14].

Das Übertragungsverhalten der phasenzyklischen Verfahren kann durch ein Laufzeitglied beschrieben werden. Die Laufzeit ist beim Resolver und beim Inductosyn reziprok

proportional der Frequenz der eingespeisten Ströme. Eine zusätzliche Verzögerung resultiert aus der geschwindigkeitsabhängigen Frequenzänderung der induzierten Spannung während der Bewegung des Meßwertaufnehmers.

3.4.4. Messung von Geschwindigkeit und Beschleunigung

Für die Realisierung zeitoptimaler Bewegungsabläufe eines Roboters sind von der Steuerung auch Informationen über die aktuelle Bewegungsgeschwindigkeit der einzelnen Achsen zu verarbeiten. Durch Differentiation des analogen Abbildsignals des Weges steht ein solches für die Geschwindigkeit zur Verfügung. Allerdings erhält man die Geschwindigkeitsinformation zeitlich erheblich verzögert, wenn das Wegsignal selbst – abhängig von der Signalverarbeitung in der Meßkette – bereits zeitlich verzögert abgebildet wird. Vor allem wirken sich bei der Differentiation des analogen Wegsignals die diesem stets überlagerten höherfrequenten Rauschspannungskomponenten äußerst störend auf das Geschwindigkeitsabbildsignal aus. Der Meßwert kann von der Störung völlig überdeckt werden. Durch die notwendige Filterung wird das Geschwindigkeitssignal am Filterausgang zusätzlich verzögert. Eine zeitoptimale Steuerung ist dann nicht möglich. Bei der Ermittlung der Geschwindigkeit aus der Beobachtung des digitalen Wegsignals in konstanten Zeitabständen treten ähnliche Probleme auf.

Aus den genannten Gründen kann für die Steuerung schneller Bewegungsabläufe eine unabhängige Geschwindigkeitsmessung erforderlich werden. Für rotatorische Bewegung werden Tachogeneratoren angeboten, die eine der Drehzahl proportionale Gleichspannung liefern. Die Welligkeit dieser Ausgangsspannung ist die das dynamische Übertragungsverhalten des Tachogenerators bestimmende Größe.

Eine direkte Messung der Beschleunigung der einzelnen Achsen ist bei den gegebenen Bewegungsmöglichkeiten eines Robotergreifers problematisch. Bei Verwendung mehrkomponentiger, piezoelektrischer Beschleunigungsaufnehmer können die in den einzelnen Achsen auftretenden Beschleunigungen nicht ohne weiteres getrennt voneinander erfaßt werden. Jedoch kann die Beschleunigung jeder Achse indirekt über die Ermittlung des zeitlichen Stromverlaufs in den jeweiligen Achsantrieben ermittelt werden.

4. Bewegungssteuerung

4.1. Grundprobleme

Die Bewegungssteuerung dient dem Ziel, die Bewegungsbefehle des Roboterprogramms auszuführen. Dabei sind drei unterschiedliche Aufgabentypen zu unterscheiden (Bild 4.1):

- Bewegung des Roboters von einer Ausgangsposition in eine Zielposition (Positionieraufgabe, Bild 4.1 a). Der Bewegungsverlauf zwischen den beiden Positionen ist meist ohne technologischen Belang, wodurch sich Freiheiten für eine Optimierung der Bewegung beispielsweise nach minimalem Zeit- oder Energieaufwand ergeben. Diese Art der Bewegung ist typisch für Handhabungsgeräte zum Beschicken von Be- und Verarbeitungsmaschinen, aber auch für Industrieroboter zum Punktschweißen.

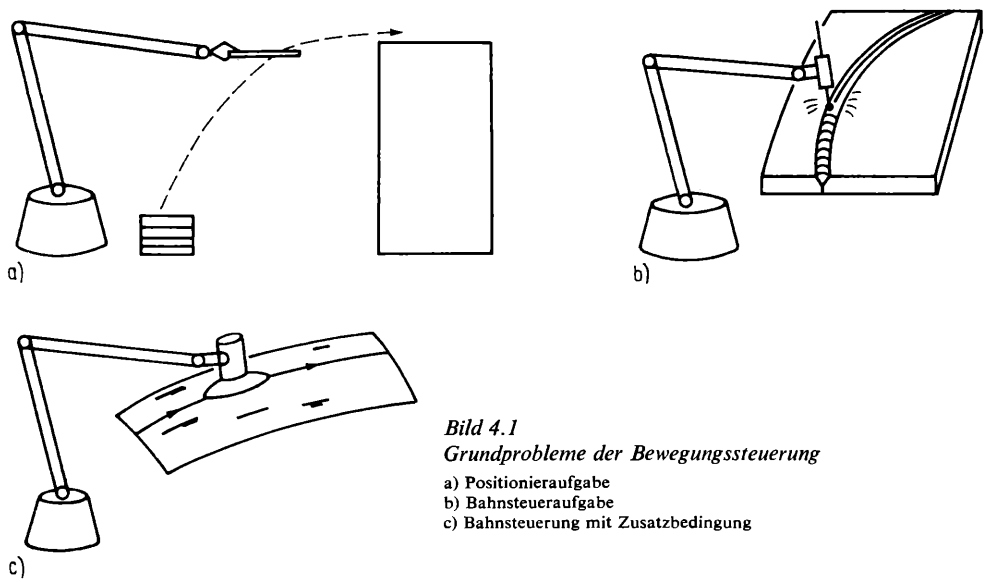


Bild 4.1
Grundprobleme der Bewegungssteuerung
a) Positionieraufgabe
b) Bahnsteueraufgabe
c) Bahnsteuerung mit Zusatzbedingung

- Bewegung des Handpunktes auf einer vorgeschriebenen Bahn mit vorgeschriebener Geschwindigkeit, wobei meistens auch die Handachsen zur Orientierung des Werkzeugs bzw. Greifers eine kontinuierliche Bewegung ausführen (Bahnsteueraufgabe, Bild 4.1 b). Hier ist der Bewegungsverlauf in allen Achsen genau vorgeschrieben. Freiheiten für eine Optimierung ergeben sich höchstens bei Beschleunigungs- und Bremsvorgängen längs der abzufahrenden Bahnkurve. Diese Bewegungsart ist typisch für Industrieroboter der ersten Generation zum Lichtbogenschweißen und Farbspritzen.
- Orientierung an einer vorgeschriebenen Bahn unter Einhaltung zusätzlicher Bedingungen. Diese Bedingungen werden durch von den Raumkoordinaten abhängige Größen formuliert, deren Wert während der Bewegung konstant gehalten oder einen Extremwert

annehmen soll. Erfaßt werden sie durch geeignete Sensoren (Bahnsteuerung mit Sensorführung, Bild 4.1 c). Solche Bewegungsaufgaben sind in der Regel überbestimmt, und es ist durch das Steuerungssystem ständig ein Kompromiß zwischen Bahntreue und Einhaltung der Zusatzbedingungen zu schließen. Typisch ist diese Art der Bewegung für Industrieroboter mit Sensorführung zur Oberflächenbearbeitung, zum Lichtbogenschweißen und für gewisse Fügevorgänge bei der Montage.

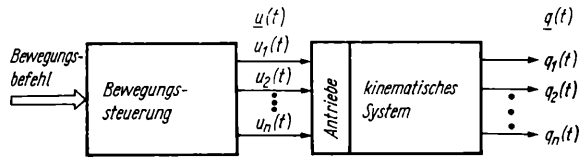


Bild 4.2
Blockschema der Bewegungssteuerung

Die Bewegungssteuerung wird durch eine Funktionsgruppe gleichen Namens realisiert, die den Antrieben Steuergrößen $\boldsymbol{u}(t) = (u_1(t), u_2(t), \ldots, u_n(t))$ so zuführt, daß die roboterspezifischen Achskoordinaten $\boldsymbol{q}(t) = (q_1(t), q_2(t), \ldots, q_n(t))$ die geforderte Bewegung ausführen (Bild 4.2). Diese erfolgt gemäß der Bewegungsgleichung des kinematischen Systems, die in allgemeiner Form

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + N(\boldsymbol{q}, \dot{\boldsymbol{q}}) = P \tag{4.1}$$

lautet [4.3]. Hier sind bei n Freiheitsgraden der kinematischen Kette

$M(\boldsymbol{q})$ eine $n*n$-Matrix aus von $\boldsymbol{q}$ abhängigen Trägheits- und Massekoeffizienten,
$N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ eine $n*1$-Matrix (Spaltenvektor) der Wechselwirkungskräfte und -momente (Zentrifugal- und Corioliseffekte) zwischen den Koordinatenbewegungen
P eine $n*1$-Matrix (Spaltenvektor) der die Bewegung verursachenden Kräfte und Drehmomente.

Die Kräfte und Momente P werden durch die Antriebe aufgebracht, die ihrerseits durch die Steuergrößen $\boldsymbol{u}$ angesteuert werden. Dabei treten in der Regel geschwindigkeitsabhängige Rückwirkungen auf, wie im Abschnitt 4.2. und im Abschnitt 3. genauer gezeigt wird. Die Modellgleichungen für die Antriebe sind daher in der Form

$$p_i = a(\dot{q}_i, u_i); \qquad i = 1, 2, \ldots, n \tag{4.2}$$

oder vektoriell

$$P = A(\dot{\boldsymbol{q}}, \boldsymbol{u})$$

anzusetzen.

Nimmt man eine additive Überlagerung der Steuerwirkungen $\boldsymbol{u}$ und der Rückwirkungen durch $\dot{\boldsymbol{q}}$ an, so entsteht aus (4.2)

$$p_i = a_i(\dot{q}_i) + b_i(u_i); \qquad i = 1, 2, \ldots, n \tag{4.3}$$

bzw.

$$P = A(\dot{\boldsymbol{q}}) + B(\boldsymbol{u})$$

und mit (4.1) für das Gesamtsystem

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + N(\boldsymbol{q}, \dot{\boldsymbol{q}}) = A(\dot{\boldsymbol{q}}) + B(\boldsymbol{u}). \tag{4.4}$$

Hier können die $n*1$-Matrizen $N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ auf der linken und $A(\dot{\boldsymbol{q}})$ auf der rechten Seite zu einer Matrix $N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) = N(\boldsymbol{q}, \dot{\boldsymbol{q}}) - A(\dot{\boldsymbol{q}})$ zusammengefaßt werden, und für das Gesamtsystem kinematische Kette mit Antrieben ergibt sich schließlich

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) = B(\boldsymbol{u}). \tag{4.5}$$

Das den Gln. (4.1) und (4.3) zugeordnete Signalflußbild ist im Bild 4.3 angegeben. Der rechte Teil entsteht dadurch, daß man zunächst Gl. (4.1) nach $\ddot{\boldsymbol{q}}$ auflöst:

$$\ddot{\boldsymbol{q}} = M^{-1}(P - N(\boldsymbol{q}, \dot{\boldsymbol{q}})). \tag{4.6}$$

Eine Kette von zwei (vektoriellen) Integratoren erzeugt aus $\ddot{\boldsymbol{q}}$ die Werte von $\dot{\boldsymbol{q}}$ und $\boldsymbol{q}$, die in ihrer Gesamtheit die Zustände des kinematischen Systems darstellen. Zwei statische Blöcke realisieren die Verknüpfungen gemäß M^{-1} und $N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ und sind entsprechend Gl. (4.6) zusammenzuschalten. Die Blockstruktur des linken Teiles von Bild 4.3 ergibt sich in offensichtlicher Weise aus Gl. (4.3).

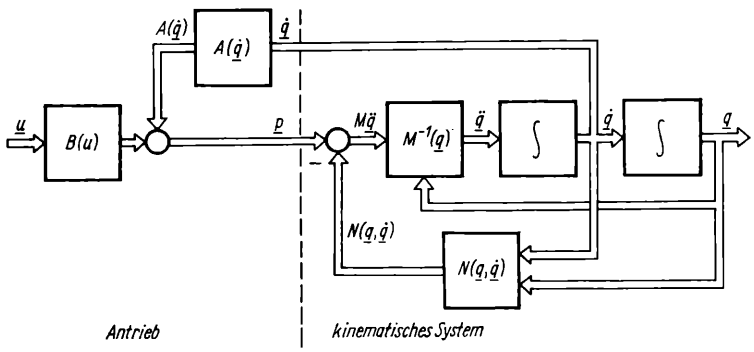


Bild 4.3. Signalflußbild des kinematischen Systems und der Antriebe

Die Aufstellung der Gl. (4.1) für konkrete Roboterkonfigurationen wird im Abschnitt 4.2. gezeigt, Modelle für die Antriebe werden im Abschnitt 3. besprochen.

Eine Grundaufgabe bei der Untersuchung kinematischer Ketten ist die Bestimmung der Bewegung der Kette bei vorgegebenem Zeitverlauf der treibenden Kräfte und Momente P, also die Lösung der Bewegungsgleichung (4.1). Für die Zwecke der Bewegungssteuerung ist die Lösung der Umkehraufgabe von besonderem Interesse.

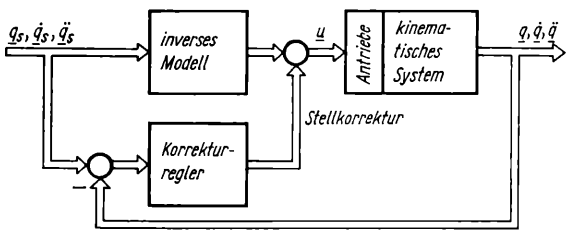


Bild 4.4
Kombinierte Steuerung und Regelung mit einem inversen Modell

Vorgegeben sind die Bewegungsverläufe, gesucht sind die Kräfte und Drehmomente, die zu ihrer Ausführung notwendig sind. Die exakte Lösung dieses Problems würde bei Einbeziehung der Modelle für die Antriebe gleichzeitig den Algorithmus für die Bewegungssteuerung im Bild 4.2 liefern. Praktisch stößt ein solches Vorgehen vorwiegend aus zwei Gründen auf Schwierigkeiten: Zum einen sind die nach Gl. (4.5) erforderlichen Berechnungen von $\boldsymbol{u}$ aus $\ddot{\boldsymbol{q}}$, $\dot{\boldsymbol{q}}$ und $\boldsymbol{q}$ sehr rechenintensiv und mit derzeit für Robotersteuerungen einsetzbaren Rechnern in Echtheit nicht realisierbar; zum anderen ist eine so präzise Modellierung des Systems kinematische Kette–Antriebe aufgrund nichterfaßbarer Störeinflüsse praktisch unmöglich. Ein Ausweg ist eine Berechnung der erforderlichen Stellgröße $\boldsymbol{u}$ über ein angenähertes inverses Modell und eine Feinkorrektur dieser Stellgröße durch ein unterlagertes Regelungssystem. Das führt auf eine Struktur nach Bild 4.4.

Da das Problem der Bewegungssteuerung als Ganzes schwierig zu lösen und die entsprechenden Algorithmen noch schwieriger zu implementieren sind, teilt man es in der Regel in zwei Teilprobleme mit klar definierter Schnittstelle: in die Bahnkurvenberechnung („Interpolation") und in die Servoregelung. Das führt auf eine Verarbeitungsstruktur nach Bild 4.5a. Ein Bahnkurvenrechner berechnet aufgrund des vorgegebenen Bewegungsbefehls die aufeinanderfolgenden Werte der zu verfahrenden Bahnkurve und gibt sie als Sollwerte auf das Servoregelungssystem, das in der Regel als Folgeregler ausgebildet ist.

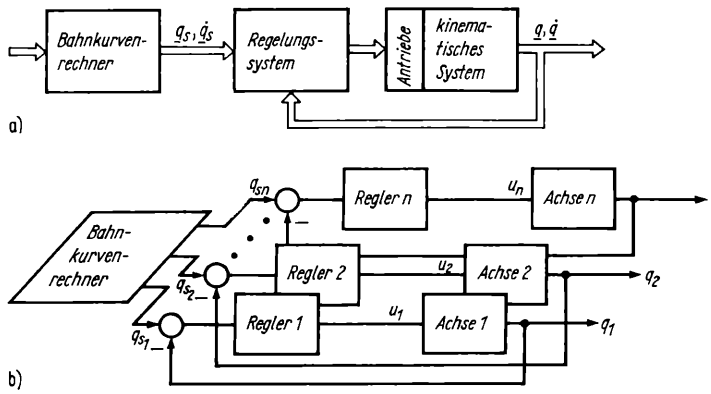


Bild 4.5
Bahnkurvenberechnung und Servoregelung
a) allgemeine Funktionsaufteilung der Bewegungssteueraufgabe
b) Ansteuerung entkoppelter Achsen

Probleme der Bahnkurvenberechnung werden im Abschnitt 4.6. behandelt.

Die Servoregelung hat die Aufgabe, die Roboterachsen entsprechend den vorgegebenen Bewegungskoordinaten zu verfahren. Entwurfs- und Bemessungskriterien für dieses funktionelle Teilsystem sind

- das Störverhalten, d.h. das weitgehende Ausregeln von a priori nicht bekannten oder erfaßten Störgrößen und Parameterveränderungen im kinematischen System und in den Antrieben, wie Reibungen, Lastvariationen und Kraftrückwirkungen durch den Bearbeitungsvorgang (beispielsweise beim Schleifen und Fügen);
- das Positionierverhalten, d.h. das zeitgünstige und überschwingfreie Erreichen von Sollpositionen vor allem bei Beschickungs-, Punktschweiß- und Montageaufgaben;
- das Folgeverhalten, d.h. das exakte Einhalten der vorgegebenen Trajektorie und der programmierten Geschwindigkeitswerte, etwa beim Lichtbogenschweißen, Farbspritzen und bei der Oberflächenbearbeitung.

Da sich diese Forderungen nicht alle gleichzeitig optimal erfüllen lassen, ergeben sich das konkrete Bemessungskriterium und die zusätzlichen Nebenbedingungen jeweils aus den Anforderungen der Klasse technologischer Aufgaben, die mit einem bestimmten Industrieroboter gelöst werden sollen.

Prinzipiell ist aufgrund der mechanischen Konstruktion von Industrierobotern die Servoregelung ein Problem der vermaschten Regelung, wobei zunächst die Werte von $q(t)$ vorgegeben sind (Bild 4.5a). Eine Verbesserung des Verhaltens der Servoregelung ist durch Einbeziehung der Ableitungen $\dot{q}(t)$ in Form einer unterlagerten Geschwindigkeitsregelung oder direkt als Sollwertvorgabe möglich. Die Lösung der Aufgabe der Servoregelung wird durch das Vorhandensein von nichtlinearen Verkopplungen zwischen den Achsen erschwert. Hier schaffen Verfahren der kombinierten Entkoppelung und Regelung Abhilfe, von denen eines im Abschnitt 4.3. behandelt wird. Bei einfachen Einsatzfällen, bei denen an die Bewegungsdynamik des Roboters keine allzu hohen Ansprüche

gestellt werden, können auch die Verkopplungen zwischen den Achsen bei der Modellbildung vernachlässigt werden. Sie gehen dann als Störgrößen in die achsspezifischen Servoregelungssysteme ein. Beide Vorgehensweisen führen zu einer Struktur nach Bild 4.5b. Die einzelnen Achsen werden dabei jede für sich nach den Sollwertvorgaben des Bahnkurvenrechners servogeregelt. Das ist bei vielen derzeit industriell ausgeführten Industrierobotern der Fall. Eine Behandlung der Einzelachsregelung findet sich im Abschnitt 4.4.

Betrachtet man die Servoregelung einer einzelnen Achse, so zeigt sich der Einfluß der Bewegungen der anderen Achsen durch additive Störkräfte [$N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ im Bild 4.3] und durch eine Beeinflussung der Parameter der Regelstrecke [$M^{-1}(\boldsymbol{q})$ im Bild 4.3]. So ist z. B. die Stellung des Roboterarms von wesentlichem Einfluß auf das effektive Trägheitsmoment der Grunddrehachse. Dieser Art Einflüsse kann durch die Verwendung von adaptiven Reglern Rechnung getragen werden, die sich einer Veränderung des Verhaltens der Regelstrecke anzupassen vermögen. Aspekte der adaptiven Regelung sind im Abschnitt 4.5. dargestellt.

Mit den vorstehenden Grundlagen werden im Prinzip die Lösungen der ersten beiden eingangs genannten Bahnsteueraufgaben erfaßt. Bei der dritten Aufgabe ist die Steuerung der Antriebe nicht nur von der gewünschten Bahnkurve und dem aktuellen Bewegungszustand des Roboters, sondern zusätzlich noch von weiteren Sensorsignalen abhängig zu machen. Solche Sensoren, die Informationen über äußere Bedingungen der Roboterbewegung, wie Nähe zu einem Hindernis, Andruckkraft während einer Bewegung, Masse des zu handhabenden Werkstücks liefern, nennt man äußere Sensoren, im Unterschied zu den inneren Sensoren, die über den Zustand des kinematischen Systems, damit auch über den aktuellen Bewegungszustand, informieren. Bei der Einbeziehung äußerer Sensoren in die Bewegungssteuerung entsteht eine weitere Rückführung (Bild 4.6). Die

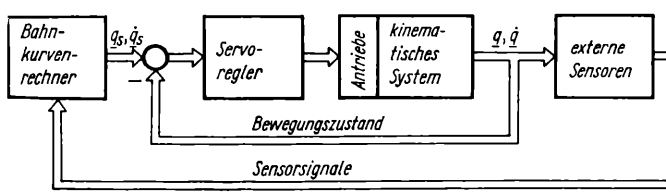


Bild 4.6
Bewegungssteuerung mit Sensorsignalverarbeitung

Sensorsignale werden dem Bahnkurvenrechner zugeleitet und bewirken hier eine Änderung der zu verfahrenden Bahnkurve so, daß die durch die Sensoren erfaßten Zusatzbedingungen eingehalten werden. Im einfachsten Fall ist dies durch eine zusätzliche Aufschaltung der Sensorsignale auf die Servoreglermischstelle im Bild 4.6 möglich; i. allg. wird die Sensorsteuerung der Servoregelung im Sinne einer Kaskadierung von Regelkreisen übergeordnet.

4.2. Modelle für das Grundgerät und die Antriebe

4.2.1. Allgemeine Ansätze

Ziel der Modellierung des Grundgeräts und der Antriebe ist es, eine geeignete Beschreibung der Regelstrecke zur optimalen Auslegung der Servoregler zu finden. Im Prinzip ist dabei die der vorliegenden mechanischen und Antriebskonfiguration entsprechende konkrete Form der Matrizen $M(\boldsymbol{q})$, $N(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in Gl. (4.1) und $A(\dot{\boldsymbol{q}}, \boldsymbol{u})$ in Gl. (4.2) zu bestimmen. Die Kinematik und die dynamischen Parameter der Hauptachsen haben bei

den am häufigsten angewendeten Konstruktionen den größten Einfluß auf das dynamische Verhalten des Grundgeräts.

Die am weitesten verbreiteten Strukturen der Hauptachsen sind die DDD- und die DDL-Strukturen (s. auch Bild 3.2). Im Bild 4.7 sind einige dieser Strukturen detaillierter dargestellt. Es ist zu erkennen, daß sich die Antriebe für Achse 1 am Grundgestell abstützen, die Antriebe für Achse 2 an Achse 1. Unterschiedliche Lösungen existieren für den Antrieb der Achse 3; er stützt sich entweder an Achse 1 (Bild 4.7b und c) oder an Achse 2 (Bild 4.7a und d) ab.

Im zweiten Fall ist die Relativlage jedes Kinematikglieds zu dem in der Kette davorliegenden eindeutig von *einem* Antrieb bestimmt. Im ersten Fall wird die Relativlage des letzten Gliedes zum vorletzten durch *zwei* Antriebe festgelegt.

Eine immer vorhandene Verkopplung besteht über das kinematische System sowie zwischen Antrieben und Kinematik, da die Antriebskraft nicht nur von der Eingangsspannung, sondern auch von der Geschwindigkeit des mit dem kinematischen System verbundenen Antriebs abhängt. Ein Antriebsgrobmodell, gültig für elektrischen und elektrohydraulischen Antrieb, ist aus den Bildern 3.12 und 3.18 erkennbar. Eingangsgrößen eines Antriebs i sind die dem Steller bzw. dem Servoventil zugeführte Spannung u_i und die auf die Antriebsachse bezogenen Geschwindigkeiten des kinematischen Systems $\dot{s}_i$ bzw. $\dot{\varphi}_i$, allgemein $\dot{q}_i$; Ausgangsgrößen sind die vom Antrieb erzeugten, auf das kinematische System wirkenden Kräfte F_i oder Momente M_i, allgemein P_i (Bild 4.7e).

In einer ersten Modellierungsstufe (Schwerpunkt: Modelle des kinematischen Systems; s. Abschnitte 4.2.2. und 4.2.3.) wird nur dieses Grobmodell des Antriebs angewendet; im Abschnitt 4.2.4. wird das Antriebsmodell detaillierter betrachtet.

Die analytischen Methoden zur Modellierung des mechanischen Grundgeräts können in zwei Hauptgruppen unterteilt werden [4.3] [4.10]:

- Methoden auf der Grundlage der Newtonschen Axiome, der Euler-Zerlegungen und des d'Alembertschen Prinzips,
- Methoden auf der Grundlage der Lagrangeschen Bewegungsgleichungen 2. Art.

Im ersten Fall werden für die einzelnen Glieder die Schwerpunkt- und Drehimpulssätze angesetzt [4.10]:

$$m\ddot{s}_S - \boldsymbol{F} = 0 \quad \text{(d'Alembertsches Prinzip)} \tag{4.7}$$

$$\boldsymbol{N} = \frac{\mathrm{d}\boldsymbol{N}}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t} \sum_i m_i \, [\boldsymbol{s}_i \dot{\boldsymbol{s}}_i] = \sum_i [\boldsymbol{s}_i \boldsymbol{F}_i] = \sum_i \boldsymbol{M}_i; \tag{4.8}$$

m Gesamtmasse des Gliedes
$\boldsymbol{F}$ Kraftvektor
$\boldsymbol{s}_S$ Lagevektor des Schwerpunktes
$\boldsymbol{N}$ Drehimpulsvektor.

Ausgehend von den kartesischen Koordinaten ergeben sich für ein Kinematikglied sechs Bewegungsgleichungen und für ein kinematisches System aus n Gliedern $6n$ skalare Bewegungsgleichungen [s. auch Gl. (3.1)]. Besitzt das System f Freiheitsgrade, dann sind $6n - f$ zusätzliche Gleichungen, die die Bindungen untereinander beschreiben, zu formulieren, was oft sehr schwierig ist. Die Aufstellung der Bindungsbedingungen wird durch die Annahme federnder Verbindung der kinematischen Glieder untereinander vereinfacht; die Zahl der Bewegungsgleichungen ist dadurch jedoch groß.

Die Anwendung von Methoden, die die Lagrangeschen Bewegungsgleichungen 2. Art nutzen, haben sich für mehrgliedrige kinematische Systeme als sehr leistungsfähig er-

wiesen [4.8] [4.9]. Nur müssen die dem Freiheitsgrad entsprechende Anzahl Zustandsgrößen ins Modell aufgenommen werden und damit für ebenso viele Koordinaten q_i die Lagrangeschen Bewegungsgleichungen 2. Art aufgestellt werden. Sehr günstig lassen sich die eingeprägten Kräfte berücksichtigen (Antriebskräfte und -momente).

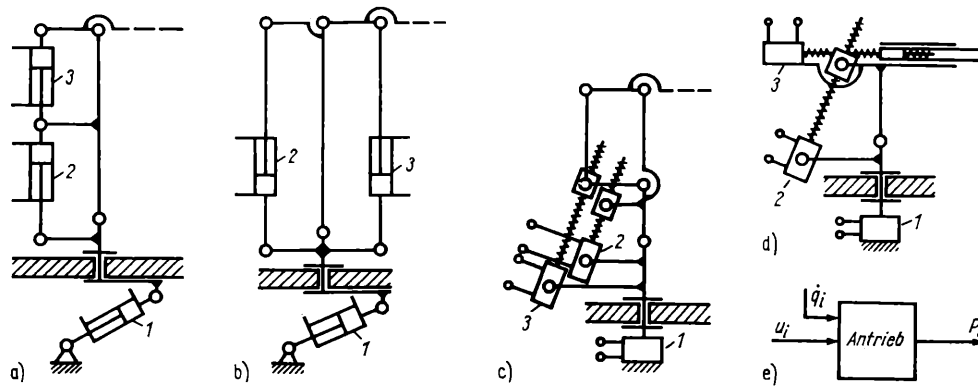


Bild 4.7. Typische Strukturen der Hauptachsen 1, 2 und 3 von Industrierobotergrundgeräten
a), b) mit hydraulischen Antrieben; c), d) mit elektrischen Antrieben; e) Grobmodell eines beliebigen Antriebs

Aus der Lagrangeschen Funktion L

$$L = E_{\text{kin}} - E_{\text{pot}} \tag{4.9}$$

ergibt sich für jede Koordinate q_i

$$\sum_j P_{ij} = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i}; \tag{4.10}$$

E_{kin} kinetische Energie des Systems
E_{pot} potentielle Energie des Systems
P_{ij} eingeprägte Kräfte und Momente, in der Koordinate q_i wirkend.

In den folgenden Abschnitten werden beide Wege der Modellierung anhand von Beispielen demonstriert.

Die z. B. in [4.3] ausführlich beschriebenen rechnergestützten Modellierungsverfahren nutzen im wesentlichen dieselben Grundlagen wie die analytischen Methoden (erweitert um die Appelschen Gleichungen und das Gaußsche Prinzip) und erleichtern die konkreten Berechnungen bei komplizierten Strukturen des kinematischen Systems erheblich.

4.2.2. Modellierung mit Hilfe der Lagrangeschen Bewegungsgleichungen 2. Art

Am Beispiel von zwei Strukturen soll die Vorgehensweise deutlich werden.

Die Struktur des ersten Beispiels ist [4.9] entnommen (s. auch Bild 4.7d). Für diese DDL-Struktur werden die Bewegungsgleichungen für den Fall

- Drehantrieb 1 bewegt,
- Schwenkantrieb 2 in Ruhe (Lage waagerecht),
- Linearantrieb 3 waagerecht bewegt

aufgestellt. Im Bild 4.8a ist das Bewegungsschema mit den interessierenden Kräften, Koordinaten und Abmessungen dargestellt:

- Erster Schritt ist die Ermittlung der potentiellen und kinematischen Energie $-E_{\text{pot}}=0$.

- Die kinetische Energie der Masse m_L setzt sich aus den Teilen rotatorische Komponente $(m_L/2)\,(s_3\dot{\varphi}_1)^2$ und translatorische Komponente $(m_L/2)\,\dot{s}_3^2$ zusammen.
- Die kinetische Energie der Masse m_3, die über die Länge l gleichmäßig verteilt ist und den Schwerpunkt S_{m3} hat, wird ebenfalls von $\dot{s}_3$ und $\dot{\varphi}_1$ bestimmt. Der translatorische Anteil beträgt $(m_3/2)\,\dot{s}_3^2$, der rotatorische $(J_3/2)\,\dot{\varphi}_1^2$. Dabei ist

$$J_3 = J(s_{m3}) + m_3\Delta s_3^2 = \frac{m_3}{12}\,l^2 + m_3\left(s_3 - \frac{l}{2}\right)^2 = m_3\left(s_3^2 + \frac{l^2}{3} - ls_3\right).$$

Es ergibt sich die Lagrangesche Funktion L zu

$$L = E_{\mathrm{kin}} = \frac{J_1}{2}\,\dot{\varphi}_1^2 + \frac{m_3}{2}\left(\dot{\varphi}_1^2\left(s_3^2 + \frac{l^2}{3} - ls_3\right) + \dot{s}_3^2 + \frac{m_L}{2}\,(s_3^2\dot{\varphi}_1^2 + \dot{s}_3^2\right). \tag{4.11}$$

Mit Hilfe von Gl. (4.10) ergeben sich über

$$\frac{\partial L}{\partial \dot{\varphi}_1} = J_1\dot{\varphi}_1 + m_3\dot{\varphi}_1\left(s_3^2 + \frac{l^2}{3} - ls_3\right) + m_L\dot{\varphi}_1 s_3^2$$

$$\frac{\partial L}{\partial \varphi_1} = 0$$

$$\frac{\partial L}{\partial \dot{s}_3} = (m_3 + m_L)\,\dot{s}_3$$

$$\frac{\partial L}{\partial s_3} = m_L\dot{\varphi}_1^2 s_3 + m_3\dot{\varphi}_1^2\left(s_3 - \frac{l}{2}\right)$$

die Bewegungsgleichungen

$$\begin{aligned} F_3 &= (m_3 + m_L)\,\ddot{s}_3 - (m_3 + m_L)\,s_3\dot{\varphi}_1^2 + m_3\,\frac{l}{2}\,\dot{\varphi}_1^2 \\ &= (m_3 + m_L)\,\ddot{s}_3 + F_F\,(s_3, \dot{\varphi}_1) \end{aligned} \tag{4.12}$$

$$\begin{aligned} M_1 &= \ddot{\varphi}_1\,(J_1 + (m_3 + m_L)\,s_3^2 + m_3\,\frac{l^2}{3} - m_3 l s_3) \\ &\quad + 2\dot{s}_3\dot{\varphi}_1\left(m_L s_3 + m_3\left(s_3 - \frac{l}{2}\right)\right) \\ &= J'(s_3)\,\ddot{\varphi}_1 + M_C\,(s_3, \dot{s}_3, \dot{\varphi}_1). \end{aligned} \tag{4.13}$$

Die jeweils zweiten Summanden der rechten Seite der beiden Bewegungsgleichungen sind die Fliehkraft F_F und das Coriolis-Moment M_C. Es ist zu erkennen, daß über sie eine Kopplung zwischen beiden Achsen besteht, die bei langsamen Bewegungen vernachlässigbar klein wird.

Die Struktur des Systems einschließlich der Abtriebe (Grobmodell) ist im Bild 4.8b dargestellt.

Die Rückwirkung von $\dot{\varphi}_1$, $\dot{s}_3$ auf den jeweiligen Antrieb ist zu erkennen [s. zum Vergleich Gl. (4.3)]; sie beeinflußt sowohl in elektrischen als auch in hydraulischen Antrieben die auf das kinematische System wirkenden Kräfte.

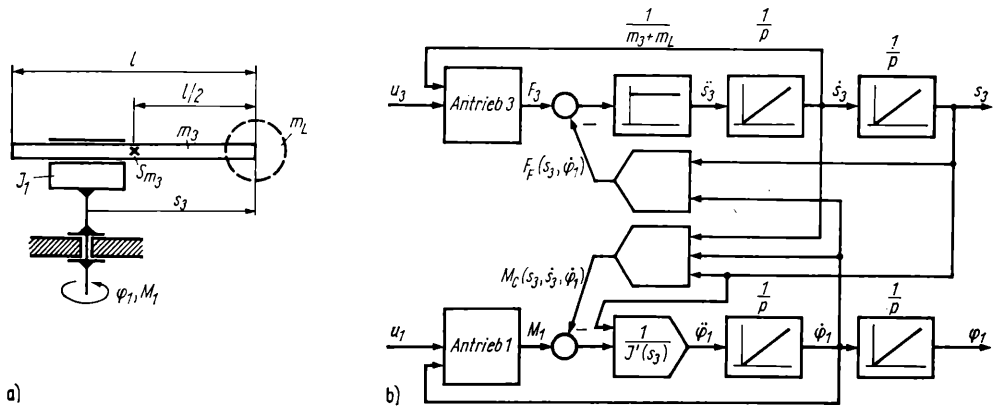


Bild 4.8. Verkopplungen der Achsen 1 und 3 in einer DDL-Struktur (ermittelt mit Hilfe des Lagrangeschen Formalismus)

a) Schema; b) Signalflußbild

Als zweites Beispiel werden die Achsen 2 und 3 eines Industrieroboters in Gelenkarmbauweise gewählt (Bild 4.7a oder b).

Für die Energiebilanz, die nach Bild 4.9a aufgestellt wird, ergeben sich unabhängig von der Antriebsgestaltung (Bild 4.9b, c und d) folgende Beziehungen (die Massen und Massenträgheitsmomente der Antriebe selbst werden dabei voraussetzungsgemäß dem kinematischen System zugeschlagen):

$$
\begin{aligned}
E_{\text{pot}1} &= l_{21}(\sin\varphi_2)\, m_2 g \\
E_{\text{pot}2} &= (l_{22}\sin\varphi_2 + l_{32}\sin\varphi_3)\, m_3 g \\
E_{\text{kin}1} &= \left(\frac{J_{S2}}{2} + \frac{m_2}{2} l_{21}^2\right)\dot{\varphi}_2^2 \\
E_{\text{kin}2} &= \frac{J_{S3}}{2}\dot{\varphi}_3^2 + \frac{m_3}{2} v_3^2 .
\end{aligned}
\tag{4.14}
$$

Die Geschwindigkeit v_3 der Masse m_3 wird wie folgt ermittelt: Aus

$$
v_3 = \sqrt{\dot{s}_{3x}^2 + \dot{s}_{3y}^2} \tag{4.15}
$$

und

$$
\begin{aligned}
s_{3x} &= l_{22}\cos\varphi_2 + l_{32}\cos\varphi_3 \\
s_{3y} &= l_{22}\sin\varphi_2 + l_{32}\sin\varphi_3 \\
\dot{s}_{3x} &= -l_{22}\sin\varphi_2\dot{\varphi}_2 - l_{32}\sin\varphi_3\dot{\varphi}_3 \\
\dot{s}_{3y} &= l_{22}\cos\varphi_2\dot{\varphi}_2 + l_{32}\cos\varphi_3\dot{\varphi}_3
\end{aligned}
$$

ergibt sich

$$
v_3^2 = \dot{s}_{3x}^2 + \dot{s}_{3y}^2 = l_{22}^2\dot{\varphi}_2^2 + l_{32}^2\dot{\varphi}_3^2 + 2 l_{22} l_{32}\cos(\varphi_2 - \varphi_3)\,\dot{\varphi}_2\dot{\varphi}_3 . \tag{4.16}
$$

Die Lagrangesche Funktion ist damit

$$L = \dot{\varphi}_2^2 \left(\frac{J_{S2}}{2} + \frac{m_2}{2} l_{21}^2 + \frac{m_3}{2} l_{22}^2 \right) + \dot{\varphi}_3^2 \left(\frac{J_{S3}}{2} + \frac{m_3}{2} l_{32}^2 \right)$$

$$+ m_3 l_{22} l_{32} \cos(\varphi_2 - \varphi_3) \dot{\varphi}_2 \dot{\varphi}_3 - g \sin\varphi_2 (m_2 l_{21} + m_3 l_{22}) - g \sin\varphi_3 (m_3 l_{32})$$

$$= \dot{\varphi}_2^2 \frac{J_2}{2} + \dot{\varphi}_3^2 \frac{J_3}{2} + J^* \cos(\varphi_2 - \varphi_3) \dot{\varphi}_2 \dot{\varphi}_3 - M'_{S2} \sin\varphi_2 - M'_{S3} \sin\varphi_3 . \tag{4.17}$$

Stimmen die Antriebsbewegungen mit den beschreibenden Koordinaten überein, hier φ_2, φ_3 (s. Bild 4.9b: Antrieb 3 stützt sich entsprechend der Festlegung für φ_3 hinsichtlich des Drehmoments an Achse 1 ab; das kommt dem Fall nach Bild 4.9d nahe), so sind die weiteren Schritte relativ einfach, da Gl. (4.17) vor der Berechnung der entsprechenden Ableitungen nicht umgeformt werden muß.

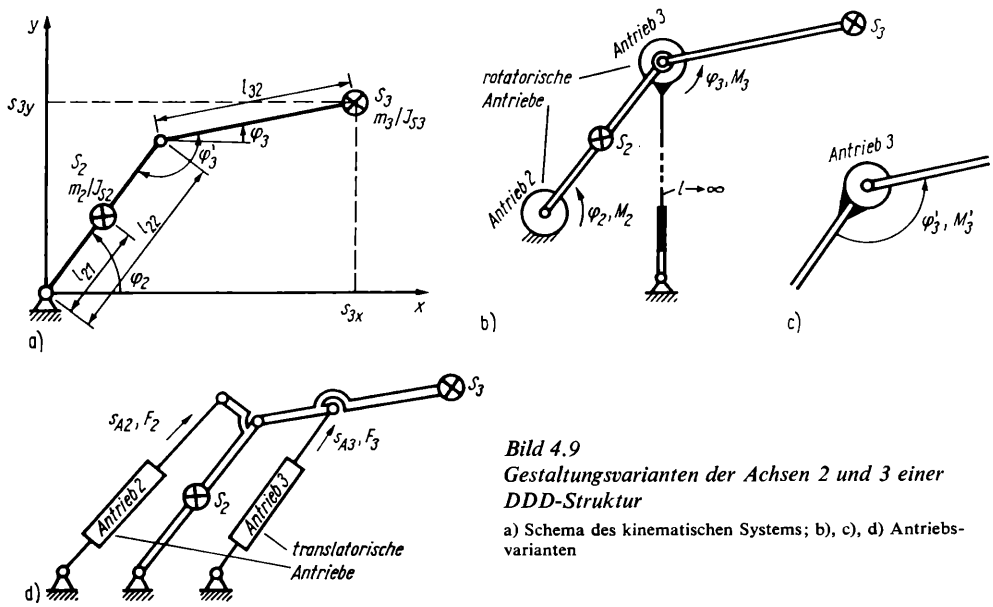


Bild 4.9
Gestaltungsvarianten der Achsen 2 und 3 einer DDD-Struktur
a) Schema des kinematischen Systems; b), c), d) Antriebsvarianten

Über die Ableitungen

$$\frac{\partial L}{\partial \dot{\varphi}_2} = \dot{\varphi}_2 J_2 + J^* \cos(\varphi_2 - \varphi_3) \dot{\varphi}_3$$

$$\frac{\partial L}{\partial \varphi_2} = J^* \dot{\varphi}_2 \dot{\varphi}_3 (-\sin(\varphi_2 - \varphi_3)) - M'_{s2} \cos\varphi_2$$

$$\frac{\partial L}{\partial \dot{\varphi}_3} = \dot{\varphi}_3 J_3 + J^* \cos(\varphi_2 - \varphi_3) \dot{\varphi}_2$$

$$\frac{\partial L}{\partial \varphi_3} = J^* \dot{\varphi}_2 \dot{\varphi}_3 \sin(\varphi_2 - \varphi_3) - M'_{s3} \cos\varphi_3$$

ergibt sich

$$M_2 = \ddot{\varphi}_2 J_2 + \ddot{\varphi}_3 J^* \cos(\varphi_2 - \varphi_3) + J^* \dot{\varphi}_3^2 \sin(\varphi_2 - \varphi_3) + M'_{S2} \cos \varphi_2$$

$$= \ddot{\varphi}_2 J_2 + \ddot{\varphi}_3 J^*_{red} + M_{F2} + M_{S2} \tag{4.18}$$

$$M_3 = \ddot{\varphi}_3 J_3 + \ddot{\varphi}_2 J^* \cos(\varphi_2 - \varphi_3) - J^* \dot{\varphi}_2^2 \sin(\varphi_2 - \varphi_3) + M'_{S3} \cos \varphi_3$$

$$= \ddot{\varphi}_3 J_3 + \ddot{\varphi}_2 J^*_{red} - M_{F3} + M_{S3}. \tag{4.19}$$

Diese Bewegungsgleichungen sind im Bild 4.10 als Signalflußbild dargestellt, wobei die Grobmodelle der Antriebe berücksichtigt wurden.

Die Kopplungen zwischen den Bewegungen sind wiederum ausgeprägt, wobei zum Fliehkrafteinfluß (M_{F2}, M_{F3}) Einflüsse der Schwerkraft (M_{S2}, M_{S3}) und der Kopplungen der Trägheitsmomente hinzukommen ($J^*_{red}\ddot{\varphi}_2$, $J^*_{red}\ddot{\varphi}_3$).

Coriolis-Momente treten bei diesem ebenen Problem nicht auf. Das wird schon dann nicht mehr gegeben sein, wenn sich der Antrieb 3, wie üblich bei einem Drehantrieb, an der Achse 2 vollständig abstützt (s. Bild 4.9c), da dann die entsprechende Koordinate nicht als Winkel φ_3 zwischen fester x-Achse und Unterarm, sondern als Winkel φ'_3 zwischen beweglichem Ober- und beweglichem Unterarm beschrieben werden muß.

Das führt auf die Beziehung

$$\varphi'_3 = \pi - \varphi_2 + \varphi_3. \tag{4.20}$$

In die Lagrangesche Funktion L nach Gl. (4.17) muß dann

für φ_3: $\varphi'_3 + \varphi_2 - \pi$

für $\dot{\varphi}_3$: $\dot{\varphi}'_3 + \dot{\varphi}_2$

für $\cos(\varphi_2 - \varphi_3)$: $-\cos \varphi'_3$

für $\sin \varphi_3$: $-\sin(\varphi_2 + \varphi'_3)$

eingeführt werden.
Das ergibt

$$L = \dot{\varphi}_2^2 \frac{J_2}{2} + (\dot{\varphi}_2 + \dot{\varphi}'_3) \frac{J_3}{2} - J^* (\cos \varphi'_3) \dot{\varphi}_2 (\dot{\varphi}_2 + \dot{\varphi}'_3)$$

$$- M'_{S2} \sin \varphi_2 + M'_{S3} \sin(\varphi_2 + \varphi'_3) \tag{4.21}$$

und führt unter Anwendung von (4.10) zu den folgenden Beziehungen:

$$M_2 = \ddot{\varphi}_2 (J_2 + J_3 - 2J^* \cos \varphi'_3) + \ddot{\varphi}'_3 (J_3 - J^* \cos \varphi'_3) + 2J^* (\sin \varphi'_3) \dot{\varphi}_2 \dot{\varphi}'_3$$

$$+ J^* \sin \varphi'_3 \dot{\varphi}'^2_3 + M'_{S2} \cos \varphi_2 - M'_{S3} \cos(\varphi_2 + \varphi'_3) \tag{4.22a}$$

$$M'_3 = \ddot{\varphi}'_3 J_3 + \ddot{\varphi}_2 (J_3 - J^* \cos \varphi'_3) - J^* (\sin \varphi'_3) \dot{\varphi}_2^2 - M'_{S3} \cos(\varphi_2 + \varphi'_3). \tag{4.22b}$$

Das zugehörige Blockschaltbild ist im Bild 4.10 angegeben. Die Beziehungen (4.22a) und (4.22b) lassen sich auf die Form

$$M_2 = \ddot{\varphi}_2 J'_2 + \ddot{\varphi}'_3 J'_3 + A(\varphi_2, \varphi'_3, \dot{\varphi}_2, \dot{\varphi}'_3) \tag{4.23a}$$

$$M'_3 = \ddot{\varphi}_2 J'_3 + \ddot{\varphi}'_3 J_3 + B(\varphi_2, \varphi'_3, \dot{\varphi}_2) \tag{4.23b}$$

bringen, wobei $\ddot{\varphi}_2$ und $\ddot{\varphi}'_3$ in beiden Gleichungen auftreten. Das führt bei der Nachbildung auf einem Rechner auf eine algebraische Schleife, die im Bild 4.10 markiert ist und die zu numerischen Schwierigkeiten führt. Um sie zu beseitigen, werden die Gln. (4.23a) und (4.23b) als lineares Gleichungssystem geschrieben

$$\ddot{\varphi}_2 J'_2 + \ddot{\varphi}'_3 J'_3 = M_{2g} = M_2 - A(\varphi_2, \varphi'_3, \dot{\varphi}_2, \dot{\varphi}'_3) \tag{4.24a}$$

$$\ddot{\varphi}_2 J'_3 + \ddot{\varphi}'_3 J_3 = M_{3g} = M'_3 - B(\varphi_2, \varphi'_3, \dot{\varphi}_2) \tag{4.24b}$$

und nach $\ddot{\varphi}_2$ und $\ddot{\varphi}'_3$ aufgelöst:

$$\ddot{\varphi}_2 = \frac{M_{2g} J_3 - M_{3g} J'_3}{J'_2 J_3 - J'^2_3} \tag{4.25a}$$

$$\ddot{\varphi}'_3 = \frac{M_{3g} J'_2 - M_{2g} J_3}{J'_2 J_3 - J'^2_3}. \tag{4.25b}$$

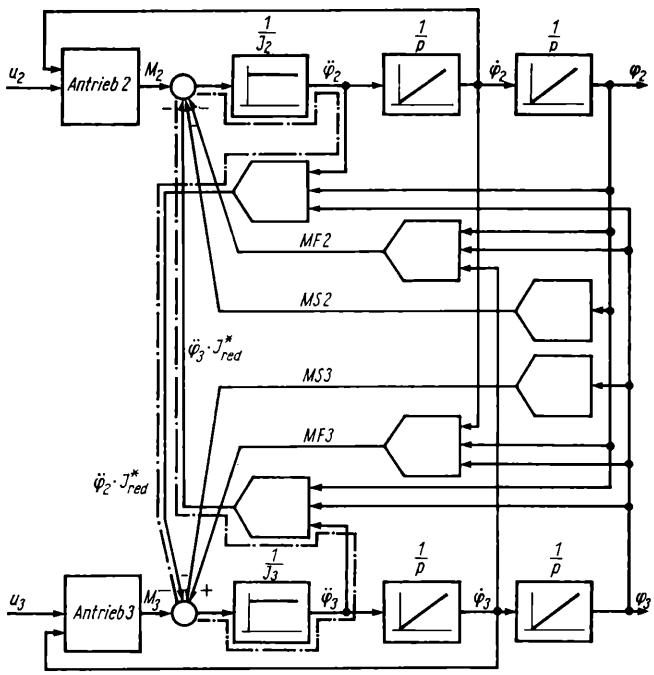


Bild 4.10
Signalflußbild der Verkopplungen der Achsen 2 und 3 einer DDD-Struktur nach Bild 4.9a und b (ermittelt mit Hilfe des Lagrangeschen Formalismus)
—.— algebraische Schleife

Hier kommen auf den rechten Seiten keine Beschleunigungen mehr vor, und die algebraische Schleife ist beseitigt. Dieses Verfahren entspricht einer Diagonalisierung der Matrix $M(\boldsymbol{q})$ und ist damit auch im Fall von mehr als zwei Dimensionen durch mathematische Standardverfahren ausführbar. Ist die Antriebsgestaltung wie im Bild 4.9d, muß die Lagrangesche Funktion L weiter umgeformt werden, damit die Koordinaten s_{A2}, s_{A3} anstelle von φ_2 und φ_3 (bzw. φ'_3) enthalten sind.

Es ist leicht zu erkennen, daß

$$\varphi_2 = f(s_{A2}),$$

$$\varphi_3 = f(s_{A2}, s_{A3})$$

ist, wobei in typischen Arbeitsbereichen der Einfluß von s_{A2} auf φ_3 vernachlässigt werden kann, so daß das Modell nach Bild 4.10 einfach an das für das System nach Bild 4.9d

angepaßt werden kann. Wesentlicher Vorteil der Ermittlung der Bewegungsgleichungen mit Hilfe des Lagrangeschen Formalismus ist auch hier, daß ein Starrkörpermodell der niedrigstmöglichen Ordnung und damit nur mit den wesentlichen Zustandsgrößen erarbeitet wird.

4.2.3. Modellierung über die allgemeine Bewegung starrer Glieder

Im Abschnitt 3.2.2. sind die Bewegungsgleichungen des massebehafteten, rotationsträgen, starren Kinematikglieds hergeleitet worden. Diese Beschreibung kann verwendet werden für kinematische Systeme aus mehreren derartigen Gliedern. Der Beschreibungsaufwand wächst mit zunehmender Zahl der Glieder schnell an, da diese Glieder in festzulegender Weise nachgiebig miteinander verbunden werden müssen. Allerdings ist die Modellierung der Federwirkungen in den Gelenken zwangsläufig gegeben, während sie bei Verwendung des Lagrangeschen Formalismus mit zusätzlichem Aufwand verbunden ist. Für eine ebene DD-Struktur nach Bild 4.7b (Achsen 2 und 3) ist das Gleichungssystem noch überschaubar. Das Antrieb-Kinematik-Schema ist im Bild 4.11a dargestellt. Der Oberarm besitzt einen, der Unterarm drei Freiheitsgrade (bedingt durch die Feder im Gelenk B, deren Verformung die Koppelkraft F_B erzeugt).

Für den Unterarm sind die folgenden skalaren Bewegungsgleichungen abzuleiten (die im Bild 4.11a fehlenden Angaben sind aus Bild 4.9a zu entnehmen):

$$\begin{aligned}
\ddot{s}_{3x} m_3 &= F_{3x} + F_{Bx} \\
\ddot{s}_{3y} m_3 &= F_{3y} - F_{By} - F_{S3} \\
\ddot{\varphi}_3 J_{S3} &= -\, l_{31x} F_{3y} + l_{32x} F_{By} + l_{31y} F_{3x} + l_{32y} F_{Bx} .
\end{aligned} \tag{4.26}$$

Die Lage der interessierenden Punkte des Unterarms B, C, D sind

$$\begin{aligned}
\Delta s_{3Bx} &= \Delta s_{3x} + l_{32y} \Delta\varphi_3 \\
\Delta s_{3By} &= \Delta s_{3y} - l_{32x} \Delta\varphi_3 \\
\Delta s_{3Cx} &= \Delta s_{3x} + l_{31y} \Delta\varphi_3 \\
\Delta s_{3Cy} &= \Delta s_{3y} - l_{31x} \Delta\varphi_3 \\
\Delta s_{3Dx} &= \Delta s_{3x} - l_{33y} \Delta\varphi_3 \\
\Delta s_{3Dy} &= \Delta s_{3y} + l_{33x} \Delta\varphi_3 .
\end{aligned} \tag{4.27}$$

Die Geschwindigkeit des Punktes C, an dem der Antrieb angelenkt ist, ist für die Rückwirkung auf den Antrieb von Interesse. Sie ergibt sich analog

$$\begin{aligned}
\dot{s}_{3Cx} &= \dot{s}_{3x} + l_{31y} \dot{\varphi}_3 \\
\dot{s}_{3Cy} &= \dot{s}_{3y} - l_{31x} \dot{\varphi}_3 .
\end{aligned} \tag{4.28}$$

Einfacher ist der Antrieb für den Oberarm. Für ihn kann die Beschreibung des Schwenkglieds im Abschnitt 3.2.4. verwendet werden. (Anstelle von F_L tritt die Koppelkraft F_B.)

Es ist

$$\ddot{\varphi}_2 \left(J_{S2} + m_2 l_{21}^2\right) = -F_2 r_A - F_{S2} r_S + F_B r_L \tag{4.29}$$

[r_A, r_S, r_L s. Gl. (3.3)].

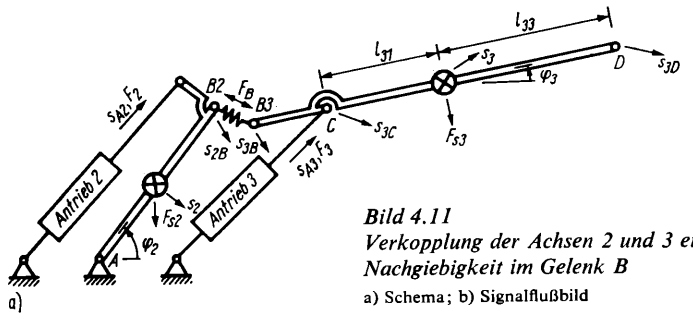
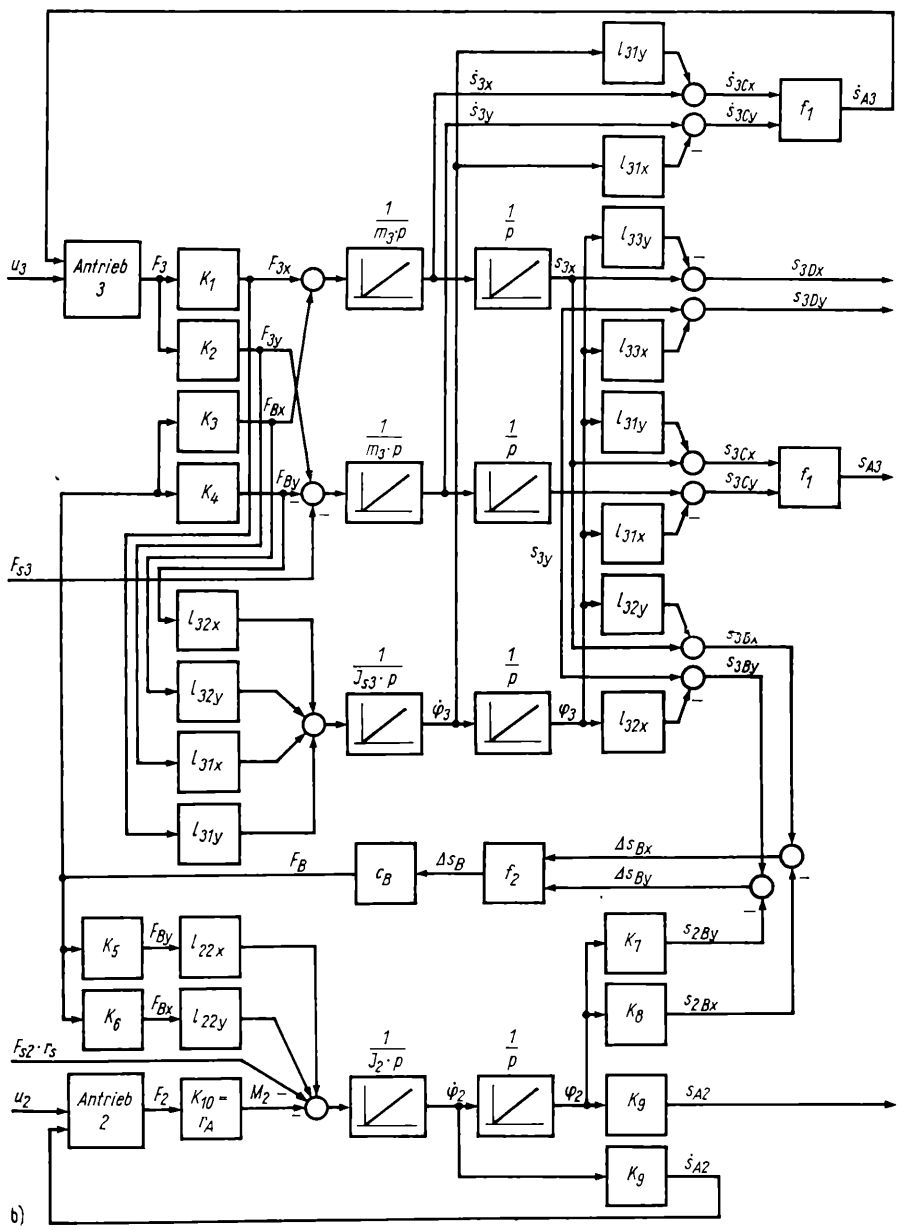


Bild 4.11
Verkopplung der Achsen 2 und 3 einer DDD-Struktur bei Nachgiebigkeit im Gelenk B
a) Schema; b) Signalflußbild



Von φ_2 allein abhängig sind s_{A2} und s_{2B}; F_B kann günstig in die x- und die y-Komponente zerlegt werden oder auch auf seine Tangentialkomponente reduziert werden. Die Koppelkraft F_B ergibt sich aus der Lagedifferenz zwischen s_{3B} und s_{2B} sowie der Nachgiebigkeit im Gelenk 3, beschrieben durch die Federkonstante c_B.

Aus Gründen der Übersichtlichkeit und der Überprüfung auf Vollständigkeit werden die angegebenen Beziehungen wiederum in einem Signalflußbild angegeben (Bild 4.11 b).

Nicht näher eingegangen werden soll auf die Beziehungen zwischen den einzelnen Kräften, Momenten, Wegen, Winkeln und ihren Komponenten. Die Faktoren K_i und l_{ijk} sind im Arbeitspunkt linearisierte Funktionen, die z. T. komplizierte trigonometrische Ausdrücke enthalten; in den Funktionen f_1, f_2 werden zwei Komponenten zu einem Gesamtweg zusammengefaßt. Aufgrund der zusätzlichen Freiheitsgrade im Gelenk B ist dieses System höherer Ordnung als die vergleichbaren Modelle im Abschnitt 4.2.2.

Vorteile der Modellierung über die allgemeine Bewegung starrer Glieder sind:

- Tatsächlich vorhandene Nachgiebigkeiten können berücksichtigt und auf relevante Stellen (z. B. Gelenke) konzentriert werden.
- Eine Erweiterung auf zusätzliche Achsen ist unter Beibehaltung des Basismodells möglich und damit relativ übersichtlich.
- Der Zeitverlauf von Gelenk- und Koppelkräften und weiterer Größen läßt sich bei Rechnungen mit dem Modell einfach verfolgen.

4.2.4. Einfache Ersatzmodelle für Kinematik und Antriebe

Die Modelle in den Bildern 4.8 b, 4.10, 4.11 b sind durch Verkopplung gekennzeichnet. Werden zusätzlich die in diesen Bildern verwendeten Grobmodelle der Antriebe (Blöcke An i) durch die der Bilder 3.12 (elektrohydraulischer Antrieb) oder 3.18 (elektrischer Antrieb) ersetzt, so ist zu erkennen, daß die Modelle bereits für zwei Achsen recht kompliziert werden. Dabei ist im Bild 3.12 nicht berücksichtigt, daß der elektrohydraulische Antrieb wesentliche Nichtlinearitäten im Servoventil enthält (s. Bild 3.11). Im folgenden soll in mehreren Stufen eine Vereinfachung so vorgenommen werden, daß Modelle entstehen, die zumindest eine Vordimensionierung der Regler mit einfachen Mitteln erlauben.

Einen Teil von Vereinfachungsmöglichkeiten bietet das kinematische System.

In den meisten Fällen ist die Rückwirkung der Lage auf das Kräftegleichgewicht gering, so daß diese in erster Näherung vernachlässigbar ist.

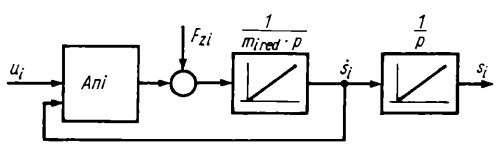


Bild 4.12
Signalfluß bei Vernachlässigung der Verkopplung und der Lageabhängigkeit der Belastungskräfte im kinematischen System

Die in den Abschnitten 4.2.2. und 4.2.3. ermittelten Verkopplungen der Achsen sind um so stärker spürbar, je größer Geschwindigkeiten und Beschleunigungen sind, denen das kinematische System unterliegt. Auf die dann notwendige Entkopplung wird im Abschnitt 4.3. eingegangen. Bei autonomer Regelung jeder Achse (s. auch Abschn. 4.1.) ist die Einwirkung des Bewegungszustands einer Achse auf die andere als Störgröße (Störkraft F_Z oder -moment M_Z) aufzufassen.

Diese beiden Vereinfachungen führen auf das Signalflußbild einer translatorischen Achse i nach Bild 4.12.

Zur Ermittlung weiterer Vereinfachungsmöglichkeiten wird der Antrieb selbst näher betrachtet.

Es werden zwei Richtungen verfolgt:

- der Bereich kleiner Aussteuerungen, in dem Begrenzungen nicht wirksam sind und linearisiert werden kann;
- der Bereich großer Aussteuerungen, in dem Begrenzungen das Verhalten wesentlich beeinflussen.

Für den elektrohydraulischen Antrieb ergibt sich bei Linearisierung das Verhalten nach Bild 3.12, das zu den Übertragungsfunktionen gemäß den Gln. (3.18) und (3.19) führt.

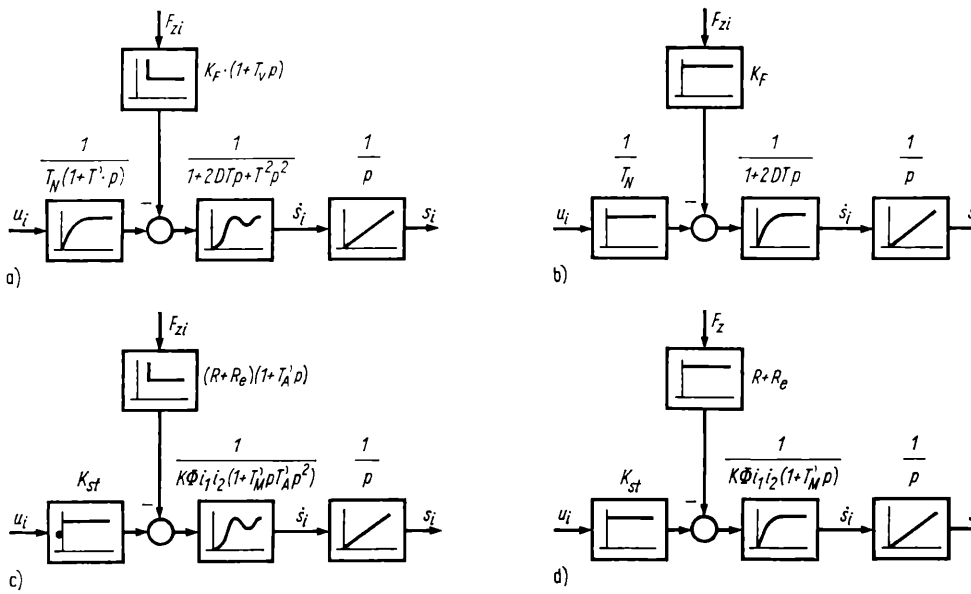


Bild 4.13. Modelle für Einzelachsen mit linearem Verhalten der Antriebe

a) elektrohydraulischer Antrieb; b) elektrohydraulischer Antrieb, Kompressibilität der Ölvolumina und Verzögerungen im Servoventil vernachlässigt; c) elektrischer Antrieb; d) elektrischer Antrieb, Steller mit Stromausgang

Das entsprechende komprimiertere Signalflußbild ist im Bild 4.13a dargestellt, wobei für die Vorsteuerstufe PT_1-Verhalten mit der Zeitkonstante T angenommen wurde.

Bei sehr kleinen Kompressionsvolumina V_1, V_2 und zusätzlicher Vernachlässigung der Verzögerung der Vorsteuerstufe des Servoventils ergibt sich das einfache Modell nach Bild 4.13b, das für viele prinzipielle Betrachtungen ausreichend genau ist.

Für den elektrischen Antrieb nach Bild 3.18 ergeben sich bei linearem Verhalten (Stellerbegrenzung nicht wirksam, Reibung vernachlässigt) die Übertragungsfunktionen nach Gl. (3.49) und (3.50). Das entsprechende Signalflußbild (s. Bild 4.13c) ist demnach Bild 4.13a für $T' \to 0$ gleich.

Für relativ kleine Induktivität $L + L_e$ (oder bei sog. Stromausgang des Stellers, mit dessen Hilfe die von den Induktivitäten hervorgerufene Verzögerung nahezu Null wird) ergibt sich $T'_A \to 0$ und ein Modell des Elektroantriebs nach Bild 4.13d, das dem Modell nach Bild 4.13b entspricht.

Zu beachten ist, daß in den Parametern $2DT$, T (Bild 4.13a und b) und T_M (Bild 4.13c und d) die reduzierte Masse $m_{i\,\mathrm{red}}$ nach Bild 4.12 enthalten ist.

Bei großen Aussteuerungen ist beim elektrischen Antrieb lediglich ein Begrenzerblock (s. Bild 3.18) vorzuschalten. Bei Stromansteuerung des Motors ergibt sich das einfache Signalflußbild nach Bild 4.14.

Im elektrohydraulischen Antrieb liegt eine Begrenzerwirkung innerhalb der Rückführschleife (s. Bild 3.12), da gemäß Bild 3.11 das Servoventil eine Begrenzung der möglichen

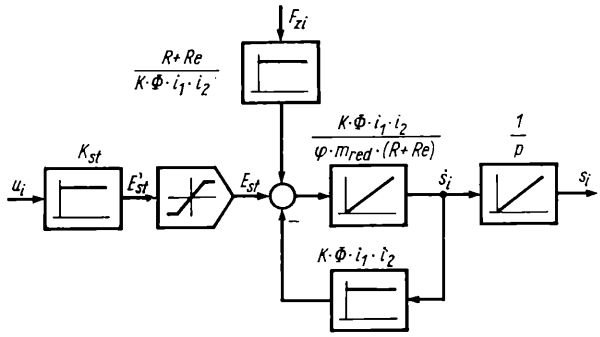
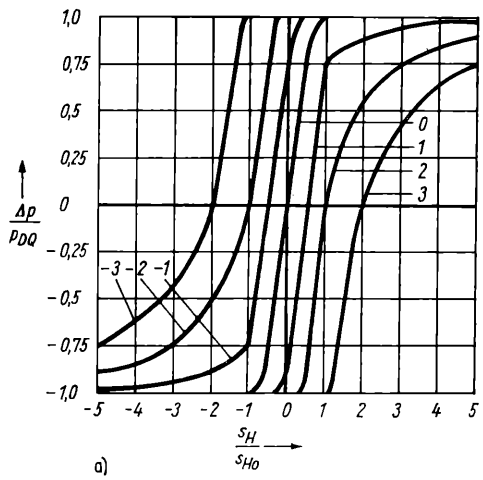
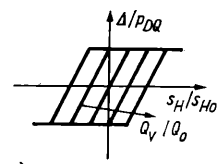
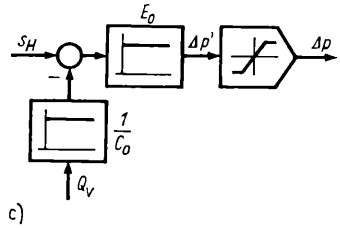
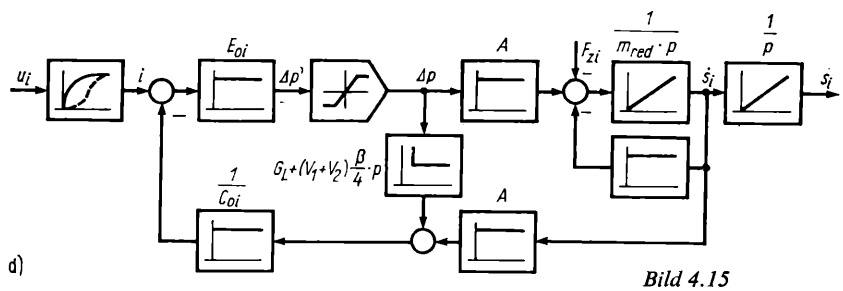


Bild 4.14
Modell einer Einzelachse mit nichtlinearem Verhalten des elektrischen Antriebs









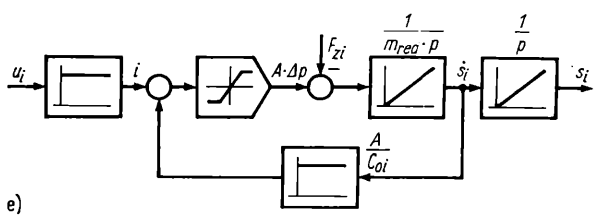


Bild 4.15
Modell einer Einzelachse mit nichtlinearem Verhalten des elektrohydraulischen Antriebs

a) theoretisches nichtlineares Kennlinienfeld der Hauptsteuerstufe eines Servoventils; b), c) Umformung in eine einfache Nichtlinearität; d) ausführliches Modell; e) Modell nach Vernachlässigung von Ölkompressibilität, Reibung und Leckö1

Druckdifferenz auf den Druckquellendruck p_{DQ} enthält. Das ist am besten zu erkennen, wenn das Kennlinienfeld im Bild 3.11b auf $\Delta p/p_{DQ} = f(s_H/s_{H0})$ mit dem Parameter Q_V/Q_0 umgestellt wird (s. Bild 4.15).

In erster Näherung kann dieses Kennlinienfeld auf die Form nach Bild 4.15b zurückgeführt werden, die wiederum leicht in das Signalflußbild nach Bild 4.15c umgeformt werden kann. Das Signalflußbild nach Bild 3.12 kann unter Nutzung von Bild 4.15c in die Form nach Bild 4.15d gebracht werden.

Unter der Voraussetzung, daß die kompressiblen Ölvolumina, der Leckleitwert G_L und die Reibungseinflüsse klein sind, ergibt sich ein einfaches, die obengenannte Begrenzung aber berücksichtigendes Modell dieses elektrohydraulischen Antriebs (Bild 4.15e).

Es ist von den Einsatzbedingungen der Antriebe und von der Zielstellung der Modellbetrachtung abhängig, ob derart vereinfachte Modelle, wie in den Bildern 4.12 bis 4.15 dargestellt, angewendet werden können. Ist dies prinzipiell möglich, muß natürlich geprüft werden, welche Stufe der Vereinfachungen noch zulässig ist.

4.3. Entkopplung der Achsen

Wie im Abschnitt 4.1. dargestellt, ist das kinematische System mit den Antrieben für die Servoregelung eine mehrdimensionale nichtlineare Regelstrecke mit nichtlinearen inneren Verkopplungen. Der Entwurf von Reglern für solche Strecken ist ein kompliziertes Problem, dessen Lösung durch Teilung in die beiden Arbeitsschritte

- Entkoppelung der Regelstrecke und
- Entwurf von Servoreglern für die entkoppelten Teilsysteme

vereinfacht und damit übersichtlicher gestaltet werden kann. Die Grundgedanken eines solchen Verfahrens, das in [4.9] ausführlich beschrieben wird, sollen nachfolgend dargelegt werden.

Ausgangspunkt ist ein vereinfachtes Modell nach Gl. (4.5):

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{U}. \tag{4.30}$$

Dieses Modell kann aus Gl. (4.5) dadurch gewonnen werden, daß entsprechend Gl. (4.3) $U_i = b_i(u_i)$ bzw. $u_i = b_i^{-1}(U_i)$ gesetzt und $\boldsymbol{U} = (U_1, U_2, \ldots, U_n)$ als neuer Steuervektor eingeführt wird. Im Signalflußbild (Bild 4.3) bedeutet das, daß dem mit $\boldsymbol{B}(\boldsymbol{u})$ gekennzeichneten Block links ein Block mit inverser Wirkung vorgeschaltet und somit eine Linearisierung des Eingangssignalwegs bis zur ersten Mischstelle links erreicht wird. Das zu Gl. (4.30) gehörende Signalflußbild der Regelstrecke ist im umrandeten Feld von Bild 4.16a dargestellt. Die Entkopplung verfolgt das Ziel, daß für $i = 1, 2, \ldots, n$ das i-te Steuersignal ausschließlich auf die Bewegung der i-ten Achse q_i wirkt. Da im Bild 4.3 die Integratoren entkoppelt sind (vgl. die Bemerkungen zur Aufstellung des Signalflußbilds im Abschnitt 4.1.), müssen die durch die Matrizen $M^{-1}(\boldsymbol{q})$ und $N^*(\boldsymbol{q}, \dot{\boldsymbol{q}})$ beschriebenen verkoppelnden Wirkungen kompensiert werden. Dazu bedarf es des Einsatzes im Prinzip gleichgearteter Kompensationselemente. Dies leistet der Ansatz

$$\boldsymbol{U} = K\boldsymbol{y} + \boldsymbol{w}, \tag{4.31}$$

wobei K eine noch zu bestimmende $n*n$-Matrix, $\boldsymbol{w}$ ein n-dimensionaler Spaltenvektor und $\boldsymbol{y}$ der neue, ebenfalls n-dimensionale Steuervektor sind.

K und w sind nun so zu bestimmen, daß das Verhalten des Gesamtsystems Strecke–Entkopplung durch die entkoppelten Gleichungen

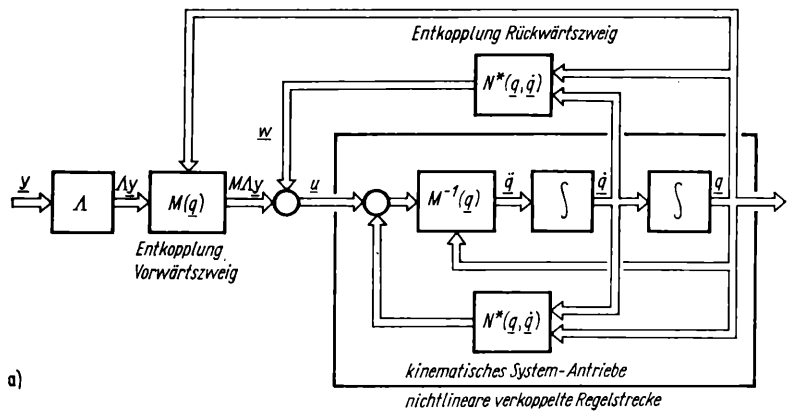
$$\ddot{q}_i = \lambda_i y_i; \qquad i = 1, 2, \ldots, n \tag{4.32}$$

oder in Matrixschreibweise

$$\ddot{\boldsymbol{q}} = \Lambda \boldsymbol{y}$$

mit der Diagonalmatrix Λ beschrieben wird. Die λ_i; $i = 1, 2, \ldots, n$ sind dabei zunächst wählbare Parameter, die eigentlich nur aus Dimensionsgründen vorzusehen sind.

Durch Einsetzen von (4.31) in (4.30) entsteht zunächst



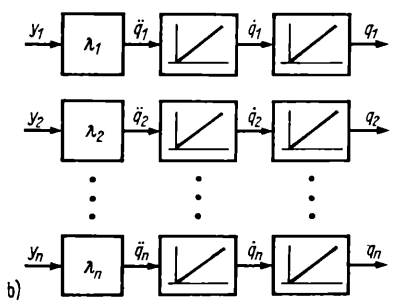


Bild 4.16
Entkopplung der Bewegungsachsen
a) Signalflußbild von verkoppelter Regelstrecke und Entkopplungsgliedern; b) Ergebnis der Entkopplung

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) = K\boldsymbol{y} + \boldsymbol{w}. \tag{4.33}$$

Wird hier

$$\boldsymbol{w} = N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{4.34}$$

gewählt, so bleibt von (4.33) die Beziehung

$$M(\boldsymbol{q})\,\ddot{\boldsymbol{q}} = K\boldsymbol{y} \tag{4.35}$$

übrig, die, nach $\ddot{\boldsymbol{q}}$ aufgelöst,

$$\ddot{\boldsymbol{q}} = M^{-1}(\boldsymbol{q})\,K\boldsymbol{y} \tag{4.36}$$

ergibt. Soll diese Beziehung die in (4.32) geforderte Form haben, so muß

$$M^{-1}(q)\,K = \Lambda \tag{4.37}$$

sein. Daraus bestimmt sich K zu

$$K = M(\boldsymbol{q})\,\Lambda. \tag{4.38}$$

Die Beziehungen (4.34) und (4.38) stellen die Bestimmungsgleichungen für $\boldsymbol{w}$ und K aus dem Ansatz (4.31) dar, der mit ihnen in die Entkoppelungsbeziehungen

$$\boldsymbol{U} = M(\boldsymbol{q})\,\Lambda \boldsymbol{y} + N(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{4.39}$$

übergeht. Das Signalflußbild (Bild 4.16a) zeigt sehr anschaulich die Wirkung des Entkopplungsverfahrens. Man erkennt, daß das Signal $\boldsymbol{w}$ gerade die Wirkung der Matrix $N^*(\boldsymbol{q}, \dot{\boldsymbol{q}})$ im Rückwärtszweig und die vorgeschaltete Matrix $M(\boldsymbol{q})$ gerade die Wirkung von $M^{-1}(\boldsymbol{q})$ innerhalb der Strecke aufhebt. Das Gesamtsystem Regelstrecke–Entkopplung verhält sich entsprechend Gl. (4.32) wie ein System von n unverkoppelten Regelstrecken mit I^2-Verhalten (Bild 4.16b).

Das zweite Teilproblem der Servoregelung besteht darin, für jede dieser Strecken einen separaten Regler anzubringen. Eine solche Einzelachsregelung ist viel einfacher zu realisieren als die Regelung eines verkoppelten Systems. Sie wird ausführlich im Abschnitt 4.4. behandelt. Hier soll als Beispiel angenommen werden, daß proportional wirkende Lageregler mit unterlagertem Geschwindigkeitsregelkreis zur Anwendung kommen. Bild 4.17 zeigt das Signalflußbild eines solchen Reglers für die i-te Achse. Man liest daraus die Beziehung

$$y_i = -c_i\dot{q}_i + d_i(q_{si} - q_i) \tag{4.40}$$

mit

$$c_i = k_{yi} \quad \text{und} \quad d_i = k_{yi}k_{Li}$$

ab, die für alle n Systeme in vektorieller Form

$$\boldsymbol{y} = -C\dot{\boldsymbol{q}} + D(\boldsymbol{q}_s - \boldsymbol{q}) \tag{4.41}$$

mit den $1*n$-Koeffizientenmatrizen C und D notiert werden kann.

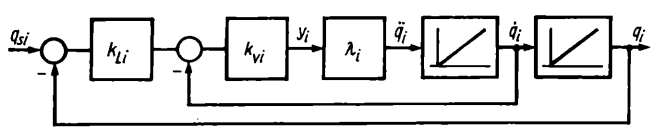


Bild 4.17
Beispiel eines Servoreglers für eine entkoppelte Achse

Das Gesamtverhalten des geschlossenen Regelungssystems der i-ten Achse ergibt sich aus (4.40) mit (4.32) zu

$$\ddot{q}_i + \lambda_i c_i \dot{q}_i + \lambda_i d_i q_i = \lambda_i d_i q_{si}. \tag{4.42}$$

Hier sind die Koeffizienten c_i, d_i und λ_i so einzustellen, daß sich ein günstiges dynamisches Verhalten ergibt. Dafür sind zahlreiche Verfahren und Untersuchungen bekannt [4.8]. Das für die Entkoppelung und Servoregelung einzusetzende Regelungssystem wird durch die Gln. (4.39) und (4.41) beschrieben. Sie lassen sich zu

$$\boldsymbol{U} = M(q)\,\Lambda\,(-C\dot{\boldsymbol{q}} + D(\boldsymbol{q}_s - \boldsymbol{q})) + N^*(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{4.43}$$

zusammenfassen. Die Beziehung (4.43) ist durch einen Algorithmus zu realisieren, der $\boldsymbol{U}$ aus $\boldsymbol{q}_s$, $\boldsymbol{q}$ und $\dot{\boldsymbol{q}}$ in Echtzeit bestimmt und den Antrieben als Stellgröße zuführt. Nachfolgend soll die konkrete Form von Gl. (4.43) für eines der im Abschnitt 4.2. behandelten kinematischen Systeme angegeben werden.

Als Beispiel wurde das System nach Bild 4.8 gewählt. Für die Durchführung der konkreten Entkopplungsschritte sind die im Bild 4.8b enthaltenen Antriebsgrobmodelle detaillierter zu beschreiben. Um überschaubare Zusammenhänge zu sichern, wird ein sehr einfaches Antriebsmodell nach Bild 3.12 mit $V_1 + V_2 \to 0$ oder nach Bild 3.18 mit $L + L_e \to 0$ und $M_R \to 0$ gewählt. In beiden Fällen ergibt sich die allgemeine Gleichung

$$P_i = K_{i1}(u_i - K_{i2}\dot{q}_i). \tag{4.44}$$

Das ergibt für das System nach Bild 4.8

$$F_3 = K_{31}(u_3 - K_{32}\dot{s}_3) \tag{4.45}$$

$$M_1 = K_{11}(u_1 - K_{12}\dot{\varphi}_1).$$

Damit ergibt sich aus Bild 4.8 das zu behandelnde verkoppelte Ausgangssystem nach Bild 4.18.

Mit (4.45) ergibt sich aus (4.12) und (4.13), aufgelöst nach den Beschleunigungen,

$$\ddot{s}_3 = \frac{K_{31}}{m_3 + m_L} u_3 - \frac{K_{31}K_{32}}{m_3 + m_L} \dot{s}_3 + s_3\dot{\varphi}_1 = \frac{m_3 l}{2(m_3 + m_L)} \dot{\varphi}_1^2 \tag{4.46}$$

$$\ddot{\varphi}_1 = \frac{K_{11}}{J'(s_3)} u_1 - \frac{K_{11}K_{12}}{J'(s_3)} \dot{\varphi}_1 - 2\dot{s}_3\dot{\varphi}_1 \frac{m_L s_3 + m_3(s_3 - l/2)}{J'(s_3)} \tag{4.47}$$

mit $J'(s_3)$ nach (4.13).

Für die Eingangsspannungen u_i wird im folgenden der Ansatz gemäß (4.31)

$$u_i = u_{i1} + K_i u_{i2} \tag{4.48}$$

gemacht.

In einer ersten Stufe, der *Entkopplungsstufe*, gilt es, den Summanden u_{i1} so zu wählen, daß das eine Teilsystem das andere nicht beeinflußt. Am einfachsten ist das realisierbar, wenn u_{11} und u_{31} so gewählt werden, daß gemäß (4.34) $\ddot{\varphi}_1$, $\ddot{s}_3$ Null werden (s. Bild 4.18). Aus Gl. (4.46) ergibt sich als Bedingung für $\ddot{s}_3 = 0$:

$$\begin{aligned} u_{31} &= K_{32}\dot{s}_3 + \frac{m_3 l}{2K_{31}} \dot{\varphi}_1^2 - \frac{m_3 + m_L}{K_{31}} s_3\dot{\varphi}_1^2 \\ &= u_{31}(s_3, \dot{s}_3, \dot{\varphi}_1) \end{aligned} \tag{4.49}$$

und aus Gl. (4.47) als Bedingung für $\ddot{\varphi}_1 = 0$:

$$\begin{aligned} u_{11} &= K_{12}\dot{\varphi}_1 + 2\dot{s}_3\dot{\varphi}_1 \frac{m_L s_3 + m_3(s_3 - l/2)}{K_{11}} \\ &= u_{11}(s_3, \dot{s}_3, \varphi_1). \end{aligned} \tag{4.50}$$

Werden die Beziehungen (4.49), (4.50) eingesetzt in (4.46), (4.47), so ergibt sich [vgl. mit (4.36)]:

$$\ddot{s}_3 = \frac{K_{31}}{m_3 + m_L} K_3 u_{32} \tag{4.51}$$

und

$$\ddot{\varphi}_1 = \frac{K_{11}}{J'(s_3)} K_1 u_{12}. \tag{4.52}$$

Das ist im Bild 4.19a dargestellt. Dieses einfache Verhalten ist dem des Teiles Ausgangssystem + Entkopplung im Bild 4.18 gleich. Die Realisierung der Entkopplung [Beziehungen (4.49), (4.50) für u_{11}, u_{31}] in einem realen System erfordert die Messung von zahlreichen Zustandsgrößen und aufwendige Berechnungen der Steuergrößen u_{11} und u_{31}.

In der zweiten Stufe gilt es, den *Vorwärtszweig* zu vereinfachen [s. Gl. (4.38)]. Er ist in vielen Fällen [s. $J'(s_3)$] nichtlinear; dies erschwert die Reglerauslegung. Die Linearisierung

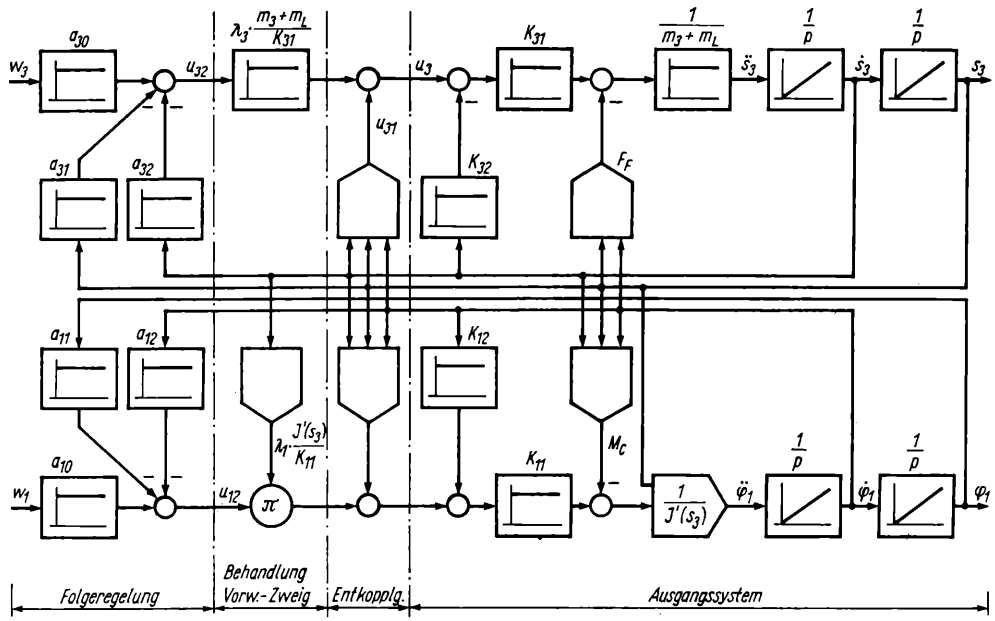


Bild 4.18. Behandlung des verkoppelten Systems nach Bild 4.8 in drei Stufen

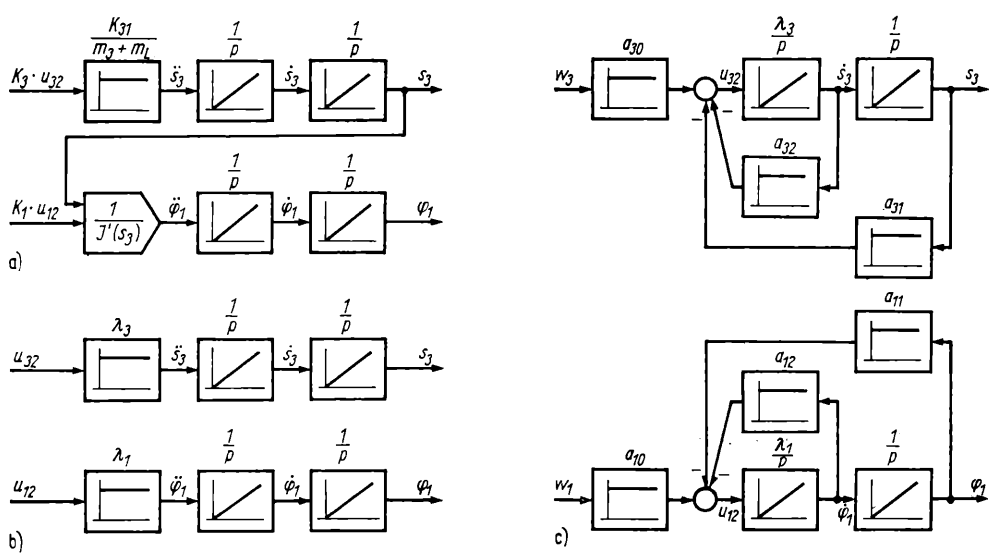


Bild 4.19. Verhalten des Systems nach Bild 4.8 nach jeder Verarbeitungsstufe gemäß Bild 4.18

a) nach der Entkopplung; b) nach Entkopplung und Behandlung des Vorwärtszweigs; c) nach Entkopplung, Vorwärtszweig und Regelung

sierung, in jedem Fall aber eine Vereinfachung, wird erreicht, wenn gemäß (4.51), (4.52) und (4.37)

$$\lambda_3 = \frac{K_{31}K_3}{m_3 + m_L}, \qquad \lambda_1 = \frac{K_{11}K_1}{J'(s_3)}$$

gesetzt wird.

Das ergibt als Korrekturgrößen für den Vorwärtszweig:

$$K_1 = \lambda_1 \frac{J'(s_3)}{K_{11}}, \qquad K_3 = \lambda_3 \frac{m_3 + m_L}{K_{31}}. \tag{4.53}$$

Damit bleiben zwei nur mit dem Faktor λ_i bewertete Integratorketten nach Bild 4.19b übrig, deren Verhalten dem des Teiles Ausgangssystem + Entkopplung + Behandlung des Vorwärtszweiges im System nach Bild 4.18 identisch sind.

Nunmehr können in der dritten Stufe die *Regler* für beide Untersysteme getrennt entworfen werden. Unter den geschaffenen Voraussetzungen können die bekannten Verfahren zur Synthese linearer Folgeregelungen ohne Einschränkungen angewendet werden. Die Vervollständigung zur Regelung [es wurde eine einfache Form wie in (4.40) bis (4.42) gewählt] ist in den Bildern 4.18 und 4.19c dargestellt (w_i Führungsgröße der i-ten Achse). In diesem Fall gilt

$$u_{i2} = w_i a_{i0} - q_i a_{i1} - \dot{q}_i a_{i2}. \tag{4.54}$$

Die Übertragungsfaktoren a_{i0}, a_{i1}, a_{i2} können den Anforderungen an das statische und dynamische Verhalten entsprechend für jede Achse getrennt frei gewählt werden, ohne daß ein Einfluß auf die andere Achse entsteht, vorausgesetzt, die Beziehungen zur Entkopplung und zur Behandlung des Vorwärtszweigs werden ausreichend genau und zu den richtigen Zeitpunkten ermittelt. Gleiches gilt natürlich auch für die innerhalb der Regelung zu berechnenden Größen für die Spannungen u_{12} und u_{32}.

Die Kompliziertheit des Systems vergrößert sich mit wachsender Zahl der verkoppelten Achsen beträchtlich (s. auch [4.9]), was zu erhöhtem Rechenaufwand für Entkopplung und Regelung führt. Eine weitere Erhöhung der Kompliziertheit des Gesamtproblems entsteht auch, wenn die Verzögerungen in den Antrieben selbst nicht mehr vernachlässigbar sind. Dann wirken die Eingangsspannungen auf höhere als die zweiten Ableitungen der Positionen nach der Zeit direkt (d. h. unverzögert) ein. Die angestrebte Integratorkette besteht dann aus mehr als zwei Integratoren. Der aufgezeigte Weg ist hier ebenfalls prinzipiell begehbar; er ist aber mit weiterer Zunahme des Aufwands für die Entkopplungs-, die Vorwärtszweig- und die Regelungsberechnungen im realen System verbunden.

Ausgehend von diesen Problemen wird in [4.11] ein Zweiebenenkonzept (eine hierarchische Regelungsstruktur) vorgeschlagen, das von einem Mehrprozessorsystem zu realisieren ist. In der unteren Ebene erfolgt bei kleiner Abtastzeit die autonome Regelung der Teilsysteme und in einer oberen, in der Koordinationsebene, bei größerer Abtastzeit die aufwendige Berechnung der Entkopplung und die Behandlung des Vorwärtszweigs. Dieser Weg erscheint in der Zukunft erfolgversprechend.

4.4. Einzelachsregelung

4.4.1. Optimales Positionieren

Das Überführen eines Industrieroboters von einer Stellung in eine andere in möglichst kurzer Zeit ist eine häufig wiederkehrende Steuerungsaufgabe. Dieses Positionierproblem soll für eine Bewegungsachse anhand des vereinfachten Modells im Bild 4.20 betrachtet

werden. Man gewinnt dieses Modell entweder nach dem im Abschnitt 4.3. behandelten Entkopplungsverfahren oder aus den im Abschnitt 4.2.4. behandelten einfachen Ersatzmodellen unter Vernachlässigung der Geschwindigkeitsrückwirkungen (vgl. die Bilder 4.14 und 4.15). Obwohl es eine sehr starke Idealisierung des dynamischen Verhaltens einer Bewegungsachse darstellt, lassen sich wesentliche Grundprobleme der Positionierung an diesem Modell studieren.

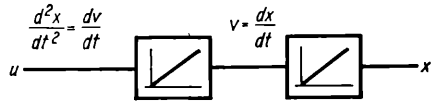


Bild 4.20
Idealisiertes Modell einer Bewegungsachse

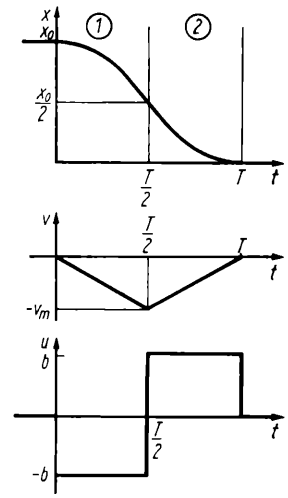


Bild 4.21
Zeitverläufe der Position x, der Geschwindigkeit v und der Steuergröße u bei einem zeitoptimalen Positioniervorgang

Die zu Bild 4.20 gehörende Bewegungsgleichung lautet

$$\frac{d^2x}{dt^2} = u. \tag{4.55}$$

Der Einfachheit halber wird dabei angenommen, daß die Beschleunigung direkt die Stellgröße der Strecke ist. Führt man die Geschwindigkeit $v = dx/dt$ ein, so entsteht aus (4.55) das Gleichungssystem

$$\frac{dv}{dt} = u(t), \qquad \frac{dx}{dt} = v(t). \tag{4.56}$$

Diese Gleichungen, die das dynamische Bewegungsverhalten der idealisierten Achse beschreiben, werden als Zustandsgleichungen für dieses System, der Vektor $v(t), x(t))^T$ als Zustandsvektor oder kurz als Zustand bezeichnet.

Die Positionieraufgabe läßt sich damit so formulieren:

Gesucht ist ein Stellgrößenverlauf $u(t)$ so, daß das durch die Gln. (4.56) beschriebene System von einem Anfangszustand $(0, x_0)^T$ in einen Endzustand (0,0) in kürzestmöglicher Zeit

$$T = \int_0^T dt = \int_{x_0}^0 \frac{dx}{v(x)}$$

übergeführt wird. Die Stellgröße (Beschleunigung) soll dabei gemäß

$$|u(t)| \leqq b \tag{4.57}$$

beschränkt sein.

Dieses Optimierungsproblem hat eine sog. „bang-bang"-Regelung zur Lösung [4.12] [4.13]. Sie besteht darin, daß bis zur Hälfte des zurückzulegenden Weges x_0 mit höchstmöglicher Beschleunigung, anschließend mit höchstmöglicher Verzögerung verfahren und abschließend die Steuergröße auf Null geschaltet wird. Bild 4.21 zeigt die zugehörigen Zeitverläufe für die Stellgröße u (Beschleunigung), die Geschwindigkeit v und den Verfahrweg x. Daß $u(t)$ in der Beschleunigungsphase und $v(t)$ im gesamten Zeitintervall negativ sind, liegt daran, daß die Bewegung von einem Ausgangswert x_0 in negativer x-Richtung zum Punkt $x = 0$ erfolgt. In den beiden Bewegungsphasen Anfahren ① und Bremsen ② gelten die folgenden Beziehungen:

	t	$u(t)$	$v(t)$	$x(t)$
① Anfahrphase	$0 \leqq t < \frac{T}{2}$	$-b$	$-bt$	$x_0 - \frac{b}{2} t^2$
② Bremsphase	$\frac{T}{2} \leqq t < T$	$+b$	$-2v_m + bt$	$\frac{b}{2}(t - T)^2$

(4.58)

Dabei errechnet sich die maximale Geschwindigkeit zu

$$v_m = \sqrt{bx_0} \tag{4.59}$$

und die Positionierdauer zu

$$T = 2\sqrt{\frac{x_0}{b}}\,. \tag{4.60}$$

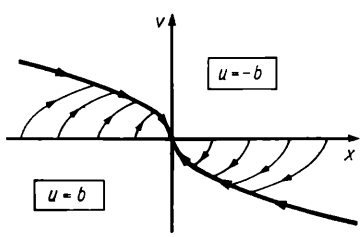


Bild 4.22
Trajektorien von zeitoptimalen Positioniervorgängen in der Zustandsebene

Trägt man die Geschwindigkeit v über dem Weg x auf, so erhält man die Zustands- oder Phasenebene des Systems. Da sich beide Größen mit der Zeit ändern, beschreibt der Endpunkt des Vektors $(v(t), x(t))^T$ eine Trajektorie in der Zustandsebene. Man kann sie aus den Zeitverläufen von $v(t)$ und $x(t)$ im Bild 4.21 konstruieren oder allgemeiner die Schar aller möglichen Trajektorien aus den Zustandsgleichungen (4.56) durch Elimination der Zeit gewinnen. Dividiert man formal (4.56a) durch (4.56b), so entsteht die Differentialgleichung

$$\frac{dv}{dx} = \frac{u}{v}, \tag{4.61}$$

deren Lösung für konstantes u

$$v(x) = \sqrt{v_0^2 + 2u(x - x_0)} \tag{4.62}$$

die Trajektorie beschreibt, die durch den Punkt $(v_0, x_0)^T$ verläuft. Damit sind die Trajektorien in der x-v-Ebene liegende Parabeln, die für $u > 0$ nach rechts, für $u < 0$ nach

links geöffnet sind. Bild 4.22 zeigt einige Trajektorien für Positionierbewegungen. Von besonderer Bedeutung ist dabei die sog. Schaltkurve, die durch die aus (4.62) zu gewinnende Beziehung

$$v = \begin{Bmatrix} -\sqrt{2bx} & \text{für } x > 0 \\ 0 & \text{für } x = 0 \\ \sqrt{2b|x|} & \text{für } x < 0 \end{Bmatrix} = -\operatorname{sgn}(x)\sqrt{2b|x|} \tag{4.63}$$

beschrieben wird. Sie trennt in der Zustandsebene zwei Gebiete voneinander, wo im Sinne der oben dargestellten optimalen Steuerstrategie die Stellgröße ihren positiven Maximalwert (unterhalb und auf der Schaltkurve) bzw. ihren negativen Maximalwert (oberhalb der Schaltkurve) annehmen muß. Für die „bang-bang"-Regelung ergibt sich damit die Funktionsgleichung

$$u = \begin{cases} b & \text{für} \quad v > -\operatorname{sgn}(x)\sqrt{2b\,|x|} \\ -b & \text{für} \quad v < -\operatorname{sgn}(x)\sqrt{2b\,|x|}, \end{cases} \tag{4.64}$$

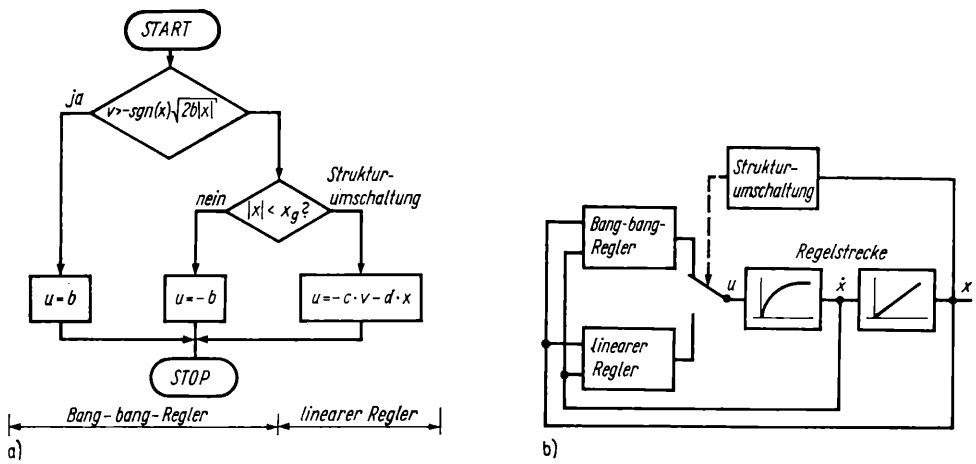


Bild 4.23. „bang-bang"-Regelung mit Strukturumschaltung
a) Steueralgorithmus; b) Signalflußbild

auf deren Basis das optimale Regelungssystem entworfen werden kann. Es wird allerdings nur dann zufriedenstellend arbeiten, wenn das dynamische Verhalten der Bewegungsachse exakt durch die Gln. (4.56) beschrieben wird und die Umschaltpunkte nach Gl. (4.64) auf das genaueste eingehalten werden. Da dies in der Praxis nicht gegeben ist, wird das Steuerungssystem nur unvollkommen funktionieren. Meist kommt es in der Umgebung des Zielpunktes zu unerwünschten Schwingungen. Um dies zu vermeiden, kann man in einer Umgebung $|x| \leqq x_g$ um den Zielpunkt den „bang-bang"-Regler nach (4.64) durch einen linearen Regler ersetzen [4.14] [4.8]. Der entsprechende Steueralgorithmus ist im Bild 4.23a dargestellt. Bild 4.23b zeigt dazu die Strukturumschaltung des Reglers im Signalflußbild. Darin ist als Beispiel das erste I-Glied des Ausgangsmodells von Bild 4.20 durch ein PT_1-Glied ersetzt worden. Das entspricht bei geeigneter Bemessung der Zeitkonstanten der Einbeziehung einer streckeninternen Geschwindigkeitsrückführung (vgl. Abschn. 4.2.4.), und das Verhalten der Regelstrecke wird nur noch in

grober Näherung durch die Gln. (4.56) beschrieben. Dennoch ist der Algorithmus nach Bild 4.23a für ein angenähert zeitoptimales Positionieren verwendbar [4.8].

Bei großen Positionierwegen ergeben sich nach Gl. (4.59) hohe Maximalgeschwindigkeiten v_m, die häufig durch die Antriebe nicht erreicht werden. In solchen Fällen ist mit Geschwindigkeitsbegrenzungen zu rechnen, und zu Gl. (4.57) kommt noch die Bedingung

$$|v(t)| = v_{max} \tag{4.65}$$

mit einer festen Maximalgeschwindigkeit hinzu. In diesem Fall vollzieht sich der Positioniervorgang in der Regel in drei Phasen:

	t	$u(t)$	$v(t)$	$x(t)$
① Anfahrphase	$0 \leqq t < t_1$	$-b$	$-bt$	$x_0 - \frac{b}{2} t^2$
	$t_1 = \frac{v_{max}}{b}$			
② Bewegung mit $v = v_{max}$	$t_1 \leqq t < t_2$			
	$t_2 = \frac{x_0}{v_{max}}$	0	v_{max}	$x_0 + \frac{v_{max}^2}{2b} - v_{max} t$
③ Bremsphase	$t_2 \leqq t < T$	b	$-v_{max} - \frac{b x_0}{v_{max}} + bt$	$\frac{b}{2}(t - T)^2$
	$T = \frac{x_0}{v_{max}} + \frac{v_{max}}{x_0}$			

(4.66)

Bild 4.24 zeigt die dazugehörigen Zeitverläufe. Für die Positionierdauer T ergibt sich

$$T = \begin{cases} 2\sqrt{\frac{x_0}{b}} & \text{für} \quad v_{max} > \sqrt{b x_0} \\ \frac{x_0}{v_{max}} + \frac{v_{max}}{x_0} & \text{für} \quad v_{max} < \sqrt{b x_0}\,, \end{cases} \tag{4.67}$$

und mit den Normierungsgrößen

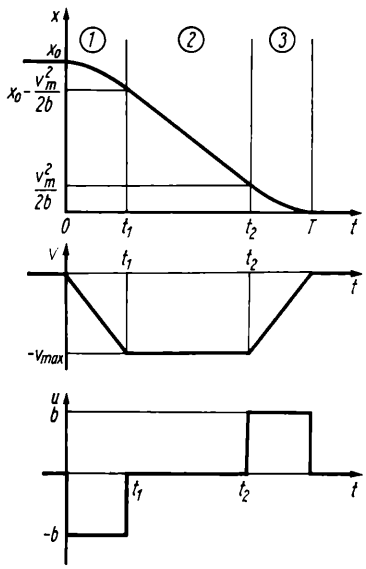
$$T_0 = \sqrt{\frac{x_0}{b}} \quad \text{und} \quad v_m = \sqrt{b x_0}$$

entsteht daraus

$$\frac{T}{T_0} = \begin{cases} 2 & \text{für} \quad v_{max} > v_m \\ \frac{v_m}{v_{max}} + \frac{v_{max}}{v_m} & \text{für} \quad v_{max} < v_m\,. \end{cases} \tag{4.68}$$

Dabei wird jeweils im ersten Fall v_{max} nicht erreicht, und die Positionierung vollzieht sich ohne Geschwindigkeitsbegrenzung (jeweils erste vier Trajektorien rechts und links vom Ursprung im Bild 4.25). Im zweiten Fall bewirkt die Geschwindigkeitsbegrenzung ein

Auftreten einer Bewegungsphase mit konstanter Geschwindigkeit (2) und damit eine Vergrößerung der Positionierdauer. Die Zustandsbewegung vollzieht sich nur in einem durch $v \leqq v_{max}$ definierten Streifen in der Zustandsebene (Bild 4.25).



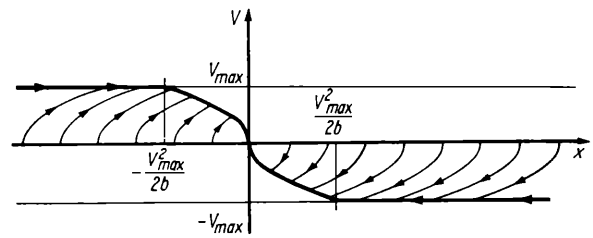


Bild 4.25. Zustandstrajektorien zu Bild 4.24

Bild 4.24
Zeitverläufe bei zeitoptimalem Positionieren mit Begrenzung der Geschwindigkeit

4.4.2. Lineare Servoregler

Die lineare Regelungstechnik ist theoretisch wie praktisch sehr weit ausgebaut. Deshalb werden lineare Regler auch für die Servoregelung von Roboterachsen bevorzugt eingesetzt, obwohl sie infolge der vorhandenen Nichtlinearitäten und der sich bei der Bewegung verändernden Parameter der Regelstrecke nicht im gesamten Verfahrbereich des Roboters optimal eingestellt werden können. Nachfolgend soll das Führungsverhalten linearer Lageregler bei Positioniervorgängen für das einfache Modell nach Bild 4.26a untersucht werden. Die Regelstrecke entspricht im Prinzip den Modellen von Bild 4.14 und Bild 4.15. Geregelt wird sie durch einen proportional wirkenden Lageregler K_L, dem ein proportionaler Geschwindigkeitsregler K_G unterlagert ist. Aus Bild 4.26 liest man direkt die Zustandsgleichungen

$$\begin{aligned} \frac{dv}{dt} &= -K_{\dot{x}} v + K_G (-v + K_L (x_w - x)) \\ \frac{dx}{dt} &= v \end{aligned} \tag{4.69}$$

ab, von denen die erste mit den Abkürzungen $K_{\dot{x}} + K_G = K_v$ und $K_L K_G = K_v K_x$ in

$$\frac{dv}{dt} = -K_v (v + K_x (x_w - x)) \tag{4.70}$$

übergeht. Das entspricht der Struktur nach Bild 4.26b. Zur Untersuchung des Positionierverhaltens wird angenommen, daß sich bei $x_w = x_0 =$ konst. der Regelkreis im eingeschwungenen Zustand ($dv/dt = 0$, $v = 0$, $x = x_w = x_0$) befindet und zum Zeitpunkt

$t = 0$ die Führungsgröße auf Null geschaltet wird. Dann ist zur Ermittlung der Zeitverläufe $x(t)$ und $v(t)$ das homogene Gleichungssystem

$$\frac{dv}{dt} = -K_v v - K_v K_x x$$
$$\frac{dx}{dt} = v \tag{4.71}$$

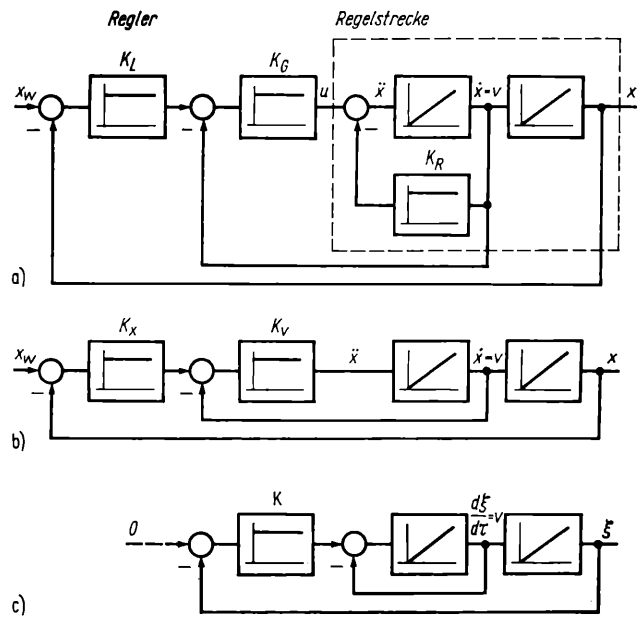


Bild 4.26
Lineare Lageregelung
a) Signalflußbild von Regler und Strecke; b) vereinfachtes Signalflußbild; c) normiertes Verhaltensmodell

oder die dazu äquivalente Differentialgleichung 2. Ordnung

$$\frac{d^2x}{dt^2} + K_v \frac{dx}{dt} + K_v K_x x = 0 \tag{4.72}$$

mit den Anfangsbedingungen

$$x(0) = x_0, \quad \left.\frac{dx}{dt}\right|_{t=0} = v(0) = 0$$

zu lösen. Mit der Normierung $\tau = K_v t$, $\xi = x/K_v$ und $K = K_x/K_v$ entsteht daraus

$$\frac{dv}{d\tau} = -v - K\xi$$
$$\frac{d\xi}{d\tau} = v \tag{4.73}$$

und

$$\frac{d^2\xi}{d\tau^2} + \frac{d\xi}{d\tau} + K\xi = 0. \tag{4.74}$$

Die zugehörige Blockstruktur ist im Bild 4.26c dargestellt. Der Lösungstyp der Differentialgleichung (4.74) ist von den Wurzeln ihres charakteristischen Polynoms $p^2 + p + K = 0$,

$$p_{1/2} = -\tfrac{1}{2} \pm \sqrt{\tfrac{1}{4} - K} \tag{4.75}$$

abhängig. Für Positionieraufgaben ist dabei der aperiodische Grenzfall $K = \frac{1}{4}$ (Doppelwurzel $p_{1/2} = -\frac{1}{2}$), bei dem der Positioniervorgang gerade ohne Überschwingen erfolgt, von Bedeutung. Hier lautet die Lösung von (4.73)

$$\begin{aligned} \xi(\tau) &= \xi(0)\left(1 + \frac{\tau}{2}\right) e^{-\tau/2} \\ v(\tau) &= -\tfrac{1}{4}\xi(0)\,\tau\, e^{-\tau/2}. \end{aligned} \tag{4.76}$$

Ist ein gewisses Überschwingen unkritisch, so kann $K > \frac{1}{4}$ gewählt werden. Dann lautet die Lösung von (4.73)

$$\begin{aligned} \xi(\tau) &= \xi(0)\left(\cos \Omega\tau + \frac{\sin \Omega\tau}{2\Omega}\right) e^{-\tau/2} \\ v(\tau) &= -\frac{K}{\Omega}\,\xi(0)\, e^{-\tau/2} \sin \Omega\tau \end{aligned} \tag{4.77}$$

mit $\Omega = \sqrt{K - \frac{1}{4}}$.

Bild 4.27 zeigt dazu die Zeitverläufe von Position, Geschwindigkeit und Beschleunigung, Bild 4.28 die entsprechenden Trajektorien in der Zustandsebene. In der letzteren kann auch das Verhalten der Beschleunigung studiert werden. Man erhält dazu aus der Gleichung von (4.73) mit $b = \mathrm{d}v/\mathrm{d}\tau =$ konst. die Gleichung

$$v = -b - K\xi \tag{4.78}$$

für die Linien konstanter Beschleunigung. Es sind parallele Geraden, deren Steigung durch $-K$ und deren Abschnitt auf der v-Achse durch $-b$ gegeben sind. Mit $b = 0$ sind sie für $K = 0{,}25$ (unterbrochene Linie) und für $K = 0{,}5$ (strichpunktierte Linie) im Bild 4.28 eingetragen.

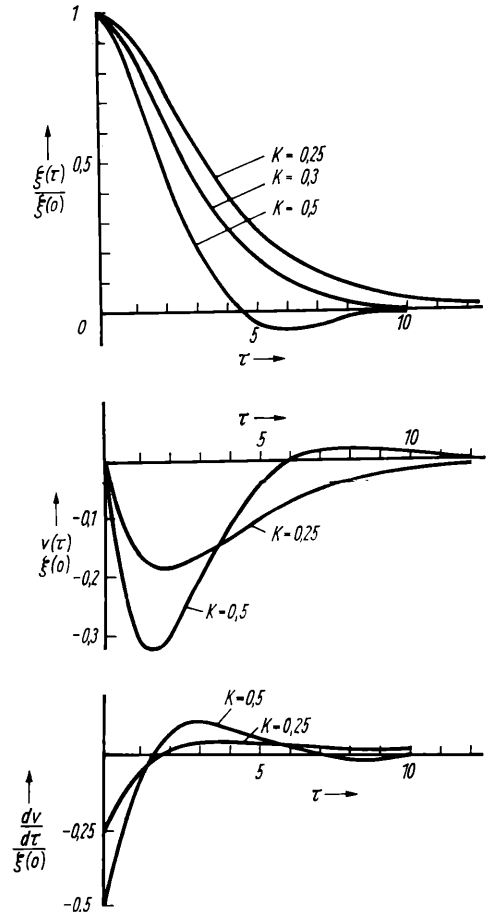
Ein Vergleich der Zeitverläufe und Trajektorien mit denen im Bild 4.21 bzw. im Bild 4.22 zeigt, daß das Verhalten der linearen Lageregelung in der gewählten Form beträchtlich vom optimalen Verhalten abweicht.

Für eine günstige Bemessung von k kann Bild 4.27 als Anhaltspunkt dienen. Da die Lösung der Gl. (4.74) erst für $\tau \to \infty$ abgeklungen ist, verwendet man als Maß für die Positionierdauer diejenige Zeit, ab der $|\xi(\tau)|$ ständig unter einer vorgegebenen Schranke, beispielsweise 1 % vom Anfangswert $\xi(0)$, bleibt. Sie soll mit $\tau_{1\%}$ bezeichnet werden. Es zeigt sich, daß man diese Zeitdauer gegenüber der beim aperiodischen Grenzfall noch deutlich senken kann, wenn ein geringes Überschwingen zugelassen ist. Man vergleiche dazu die Verläufe von $x(t)$ im Bild 4.27. Die Überschwingweite s ist durch das Minimum von $\xi(\tau_m)$ aus Gl. (4.77) an der Stelle $\tau_m = \pi/\Omega$ gegeben. Es berechnet sich folglich zu

$$s = e^{-\pi/2\Omega} \tag{4.79}$$

und wird meist in Prozent angegeben ($s_{\%} = 100\ s$). Bild 4.29 zeigt die Abhängigkeit der 1-%-Positionierdauer $\tau_{1\%}$ und der Überschwingweite $s_{\%}$ in Abhängigkeit von K. Man erkennt, daß günstige Einstellwerte für K im Bereich 0,25 bis 0,35 liegen.

Bei den Betrachtungen zur Modellbildung in den Abschnitten 3. und 4.2. ist mehrfach auf immer vorhandene Begrenzungen im Regler und in der Strecke hingewiesen worden. Um den Einfluß solcher Begrenzungen zu untersuchen, werden die beiden linearen Regler



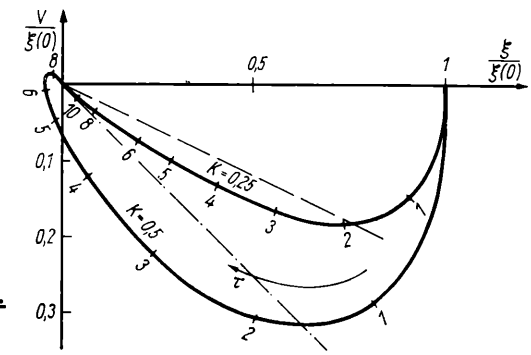


Bild 4.28. Zustandstrajektorien beim Positionieren mit linearer Regelung

Bild 4.27
Zeitverläufe von Position, Geschwindigkeit und Beschleunigung beim Positionieren mit linearer Regelung

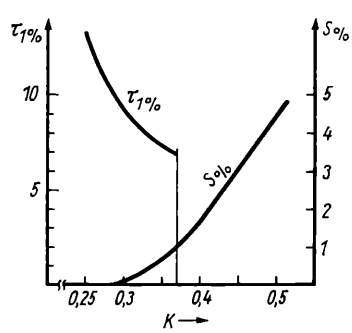


Bild 4.29. Abhängigkeit der 1-%-Positionsdauer und der Überschwingweite von k

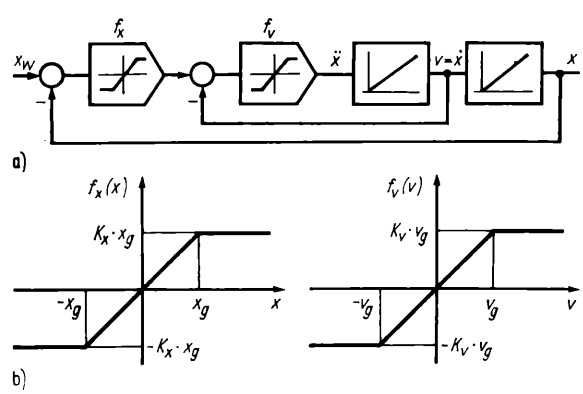


Bild 4.30. Lineare Lage- und Geschwindigkeitsregler mit Begrenzung
a) Signalflußbild; b) zur Festlegung der Funktionen f_x und f_v

im Bild 4.26a durch solche mit Begrenzungen ersetzt und die Reibung vernachlässigt ($K_x = 0$). Das entstehende Signalflußbild (Bild 4.30a) ist dem von Bild 4.26b strukturäquivalent. Damit gelten anstelle von (4.69) die Zustandsgleichungen

$$\frac{dv}{dt} = f_v(-v + f_x(x_w - x))$$
$$\frac{dx}{dt} = v, \tag{4.80}$$

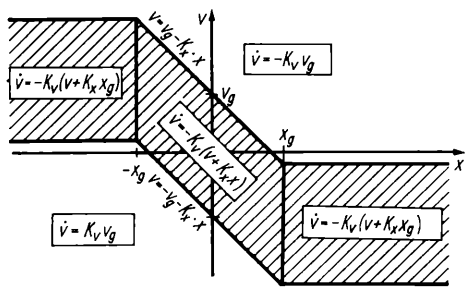


Bild 4.31
Gebietsaufteilung der Zustandsebene durch die Begrenzungen

die für einen Positioniervorgang wieder mit $x_w = 0$ und den Anfangsbedingungen $x(0) = x_0$ und $v(0) = 0$ zu lösen sind. Die beiden Funktionen f_x und f_v sind mit den entsprechenden Bezeichnungen im Bild 4.30b angegeben. Da sie stückweise definiert sind, wird die Zustandsebene in verschiedene Gebiete aufgeteilt, in denen die erste der beiden Gleichungen (4.80) jeweils spezielle Formen annimmt. Im Bild 4.31 sind diese Gebiete mit den entsprechenden Formen der Gleichung für $dv/dt = \dot{v}$ dargestellt. Außerhalb des

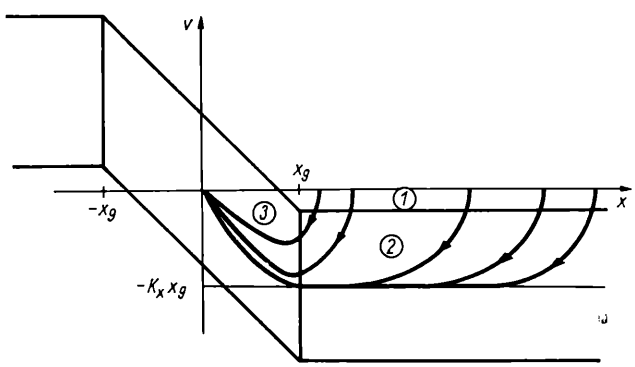


Bild 4.32
Zustandstrajektorien für Positioniervorgänge bei Begrenzungen

schraffierten Streifens sind beide Regler übersteuert, innerhalb arbeitet der Geschwindigkeitsregler (f_v) in seinem linearen Bereich. Der Lageregler (f_x) arbeitet im mittleren Gebiet und ist für $|x| > x_g$ übersteuert. Durch abschnittsweises Lösen der Differentialgleichungen (4.80) erhält man die Zeitverläufe bzw. Trajektorien des Positioniervorgangs. Typische Trajektorien sind im Bild 4.32 dargestellt. Ist $|x_0| > x_g$, so erfolgt der Positioniervorgang in drei Phasen:

(1) Beide Regler sind übersteuert; die Bewegung erfolgt mit maximal möglicher Beschleunigung; die Geschwindigkeit nimmt linear zu, die Position mit t^2 ab. Die Trajektorien sind Parabeln (Bild 4.22).

(2) Der Geschwindigkeitsregler arbeitet in seinem Proportionalbereich, die Geschwindigkeit strebt exponentiell dem Endwert $-K_x x_g$ zu; die Abnahme der Position erfolgt anfangs exponentiell, später linear.

(3) Beide Regler arbeiten in ihrem Proportionalbereich. Es gelten dabei die Gln. (4.71), deren Lösungen bereits diskutiert wurden.

Berücksichtigt man, daß der Begrenzung x_g des Lagereglers in der Praxis einige Millimeter entsprechen, so ist zu erkennen, daß der Positioniervorgang überwiegend bei übersteuertem Lageregler, d.h. praktisch bei offenem Lageregelkreis verläuft. Erst in der letzten Phase des Bremsens und Einfahrens in die Position werden beide Regler wirksam. Um die nahezu unkontrollierte Bewegung der Achse in den ersten beiden Phasen zu vermeiden, empfiehlt es sich, durch eine geeignete Sollwertaufbereitung und -führung Sollwertsprünge auszuschließen und so Lage- und Geschwindigkeitsregler während des gesamten Positioniervorgangs im Proportionalbereich zu halten. Das wird im nachfolgenden Abschnitt besprochen.

4.4.3. Führungsverhalten des Antriebs

Um den Antrieb innerhalb der gesamten Bewegungsphase trotz vorhandener Stellgrößenbeschränkungen im Regelbereich betreiben zu können, ist es erforderlich, eine Führungsgröße bereitzustellen, der der Antrieb jederzeit zu folgen vermag (Nachlaufregelung). Das ist insbesondere dann wichtig, wenn beim Positionieren mehrere Achsen verfahren werden und sich der Greiferendpunkt auf einer definierten (wenn auch oft nur grob tolerierten) Bahn bewegen soll. Die dabei prinzipiell gültigen Zusammenhänge zwischen Kennwerten der Regelstrecke und erforderlicher Führungsgröße sollen am einfachen linearisierten Antriebsmodell nach Abschnitt 4.2.4. bzw. Bild 4.13 erläutert werden. Das Verhalten des Antriebs ist durch folgende Übertragungsfunktionen zu beschreiben:

Führung

$$G_x(p) = \frac{x(p)}{u(p)} = \frac{K_u}{1 + T_M p} \frac{1}{p} \tag{4.81}$$

Störung

$$G_F(p) = \frac{x(p)}{F(p)} = \frac{K_F}{1 + T_M p} \frac{1}{p}. \tag{4.82}$$

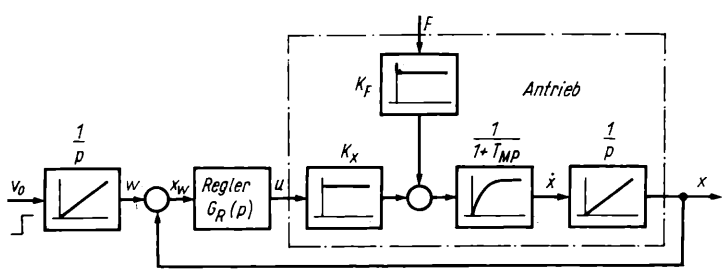


Bild 4.33 Vereinfachtes Signalflußbild der Lageregelung

Die speziellen Faktoren K_u, K_F und T_M sind gemäß Abschnitt 4.2.4. zu bestimmen. (Beachte: Die Größen können arbeitspunktabhängig sein.)

Zur Vermeidung von Übersteuerungen sind anstelle von Eingangssprungfunktionen stetige (bzw. stückweise stetige) Führungsfunktionen dem Regelkreis aufzuschalten. Die einfachste Form ist eine Rampenfunktion $w(t) = v_0 t$. Die Gesamtübertragungsfunktion

$G_w(p) = x(p)/w(p)$ mit einem einfachen P-Regler $[G_R(p) = K_R]$ berechnet sich nach Bild 4.33 zu

$$G_w(p) = \frac{x(p)}{w(p)} = \frac{1}{1 + \frac{1}{K_u K_R} p + \frac{T_M}{K_u K_R} p^2} = \frac{1}{1 + T_1 p + T_2^2 p^2}$$

$$= \frac{1}{1 + \frac{2D}{\omega_0} p + \frac{1}{\omega_0^2} p^2}; \tag{4.83}$$

ω_0 Eigenfrequenz
D Dämpfung.

Der Zeitverlauf der Regelgröße $x(t)$ für $w(t) = v_0 t$ ergibt sich aus der Lösung der zugehörigen Differentialgleichung

$$x(t) = v_0 \left[t - \frac{2D}{\omega_0} - e^{-D\omega_0 t} \left(\frac{1 - 2D^2}{\omega_0 \sqrt{1 - D^2}} \sin\sqrt{1 - D^2}\, \omega_0 t - \frac{2D}{\omega_0} \cos \sqrt{1 - D^2}\, \omega_0 t \right) \right]. \tag{4.84}$$

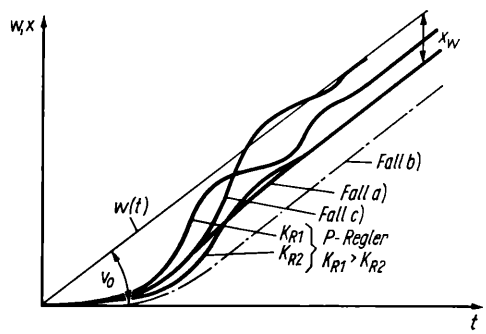


Bild 4.34
Qualitativer Verlauf $x(t)$ für rampenförmiges Eingangssignal bei unterschiedlichen P- (PD-) Rückführungen

Mögliche Antwortfunktionen sind im Bild 4.34 dargestellt. Die bleibende stationäre Regelabweichung beträgt

$$x_w = x - w = -\frac{2D}{\omega_0} v_0 = -\frac{v_0}{K_u K_R}. \tag{4.85}$$

Sie steigt mit zunehmender Rampensteilheit und sinkt mit wachsender Reglerverstärkung. K_R kann zur Erzielung kleiner Abweichungen jedoch nicht beliebig vergrößert werden. Das zeigt eine Betrachtung der Dämpfung des Einschwingvorgangs nach Gl. (4.83):

$$D = \frac{T_1}{2T_2} = \frac{1}{2\sqrt{K_u K_R T_M}}. \tag{4.86}$$

Sie sinkt mit steigendem K_R; der Übergangsvorgang verläuft schwingend; die stationäre Phase wird später erreicht. Ein schnelles Einschwingen wird mit $D = 0{,}7$ erzielt. Für gegebenen Antrieb (K_u, T_M) ist die dazu erforderliche Reglerverstärkung

$$K_R = \frac{1}{2K_u T_M}, \tag{4.87}$$

womit sich eine bleibende Folgeabweichung x_w

$$x_w = x - w = -2T_M v_0 \tag{4.88}$$

einstellt. Demgemäß darf für eine vorgegebene zulässige Abweichung $x_{w\,zul}$ der Rampenanstieg der Führungsgröße v_0 den Wert

$$v_{0\,max} = \frac{x_{w\,zul}}{2T_M} \tag{4.89}$$

nicht überschreiten. Die Zeitkonstante des Antriebs legt so die größtmögliche Änderungsgeschwindigkeit der Führungsgröße fest. Bei einer Sättigungsgrenze des Reglerausgangs von $\pm U_0$ ergibt sich $x_{w\,zul} = U_0/K_R$.

Bei Einführung eines zusätzlichen D-Anteils ergeben sich folgende Veränderungen:

a) zusätzlicher D-Anteil im Regler (PD-Regler)

$$u = K_R\left[(w - x) + T_V \frac{d\,(w - x)}{dt}\right]$$

$$x_w = -\frac{v_0}{K_u K_R}, \qquad D = \frac{1 + K_u K_R T_V}{2\sqrt{K_u K_R T_M}}. \tag{4.90}$$

Während sich gegenüber reinem P-Regler die statische Folgeabweichung nicht verändert, steigt die Dämpfung D, was bei gleichem Gesamtdämpfungsfaktor [s. Gl. (4.86)] eine Vergrößerung von K_R zuläßt.

b) Rückführung der Regelgröße x über PD-Glied auf die Summationsstelle

$$u = K_R\left[(w - x) + T_V \frac{dw}{dt}\right]$$

$$x_w = -\frac{1 + K_u K_R T_V}{K_u K_R} v_0, \qquad D = \frac{1 + K_u K_R T_V}{2\sqrt{K_u K_R T_M}}. \tag{4.91}$$

Die statische Abweichung vergrößert sich. Gleichzeitig tritt eine Dämpfungserhöhung auf.

c) Führungsgröße wird über PD-Glied auf das Summationsglied geschaltet

$$u = K_R\left[(w - x) + T_V \frac{dx}{dt}\right]$$

$$x_w = -\left(\frac{1}{K_u K_R} - T_V\right) v_0, \qquad D = \frac{1}{2\sqrt{K_u K_R T_M}}. \tag{4.92}$$

Durch die Wahl von $T_V = 1/K_u K_R$ kann der Folgefehler zu Null (für $T_V > 1/K_u K_R$ sogar positiv) gemacht werden. Die Dämpfung bleibt gegenüber P-Regler erhalten. Es ist zu beachten, daß D-Anteile leicht zur Übersteuerung der Reglerbaugruppe führen und zu den berechneten Folgefehlern noch die Abweichungen infolge wirksamer Störkräfte zu addieren sind.

$$\Delta x = \frac{K_F}{K_u K_R} F, \tag{4.93}$$

die, sofern die Belastung variabel ist, mit D-Anteilen nicht zu kompensieren sind.

Eine Verbesserung des Führungsverhaltens besteht in einem dem Regelkreiseingang vorgeschalteten Korrekturglied G_K, das als Übertragungsfunktion den Reziprokausdruck der Führungsübertragungsfunktion $G_w(p)$ besitzt. Die bei theoretischer Betrachtung ideale Maßnahme ($G_{ges} = G_K G_w = (1/G_w)\, G_w \equiv 1$) der vollständigen Kompensation der Regelkreisdynamik hat jedoch praktische Grenzen:

- In der Übertragungsfunktion des Regelkreises

$$G_w(p) = \frac{x(p)}{w(p)} = \frac{z(p)}{N(p)} = \frac{1 + T_{v1}p + T_{v2}^2 p^2 + \ldots + T_{vn}^n p^n}{1 + T_1 p + T_2^2 p^2 + \ldots + T_n^n p^n} \tag{4.94}$$

  ist das Nennerpolynom in der Regel höheren Grades als das Zählerpolynom ($m > n$). Im Korrekturglied muß demzufolge das Zählerpolynom höherer Ordnung als das Nennerpolynom sein, was technisch nicht realisierbar ist (maximal Gleichheit). Damit beschränkt sich diese Methode auf die Kompensation nur einiger, für den Regelvorgang ungünstiger Zeitkonstanten.
- Bezüglich der zu bildenden Abweichungen ($\dot{w}$, $\ddot{w}$ usw.) muß die Führungsfunktion stetig (differenzierbar) sein.
- Indem der Reglereingang neben der Regelabweichung zusätzlich mit den zeitlichen Ableitungen der Führungsgröße beaufschlagt wird, kann er rasch übersteuert werden, so daß praktisch der entsprechende D-Anteil nicht mehr wirksam ist.
- Die tatsächlichen Antriebsmodelle können komplizierter sein und aussteuerungsabhängige Parameter haben.

Die in Gl. (4.93) berechnete Abweichung Δx infolge äußerer Belastungskräfte F kann stationär nur durch I-Anteile im Regler beseitigt werden. (Beachte: Ein reiner I-Regler bedeutet strukturinstabilen Regelkreis, deshalb Verwendung von PI- bzw. PID-Varianten.) In [4.15] wird anschaulich gezeigt, wie viele I-Anteile installiert sein müssen, um für spezielle Eingangsfunktionen $x_w \to 0$ zu sichern. Für $w(t) = b/2t^2$ (parabelförmiger Verlauf, vgl. Bild 4.21) sind z. B. drei I-Anteile in der Übertragungsfunktion des offenen Regelkreises zur Vermeidung stationärer Nachlauffehler erforderlich.

I-Komponenten im Regler können mit folgenden Nachteilen verbunden sein:

- Sie begünstigen die Schwingneigung (deshalb oft relativ große Nachstellzeiten notwendig; stationärer Zustand wird entsprechend spät erreicht).
- Überschwingfreies Einlaufen in die Position wird erschwert (deshalb wird der I-Anteil oft begrenzt bzw. in der Verfahrphase sogar vollständig abgeschaltet).

In den bisherigen Betrachtungen wurde nur die stationäre Abweichung betrachtet, da sie mit einfachen Gleichungen beschreibbar ist. Die dynamischen Abweichungen – zu ermitteln über die Lösung von (teilweise nichtlinearen) Differentialgleichungen – können die hier berechneten Werte oft wesentlich überschreiten. Geht man jedoch davon aus, daß beim Positioniervorgang das Verfahren mit konstanter Geschwindigkeit [entspricht $w(t) = v_0 t$] dem größten Teil der Bewegungsphase entspricht (vgl. Bild 4.35) und dort der stationäre Zustand erreicht wird, können die angegebenen Gleichungen als grobe Näherung bei Festlegung maximaler Änderungsgeschwindigkeiten der Führungsgröße zur Vermeidung des Arbeitens im Sättigungsbereich angesehen werden.

Für praktische Anwendungen werden für Positionieraufgaben aus den abgespeicherten Start- und Zielpositionen Führungsfunktionen nach Bild 4.35 generiert. Anstelle von Zeitfunktionen $w(t)$ wird häufig die Führungsgröße w als Funktion des Abstands der Regelgröße x von der Startposition (Anfahren) bzw. von der Zielposition (Abbremsen)

vorgegeben. Zweckmäßigerweise erfolgt die Bestimmung des aktuellen Wertes von w nicht durch ständige numerische Berechnung, sondern (besonders für Anlauf und Bremsen) durch abgespeicherte Tabellenwerte (z.B. Bremsen: Differenz Zielposition X2 – Istwert x adressiert eine Speicherzelle, in der der zugehörige Wert für w steht).

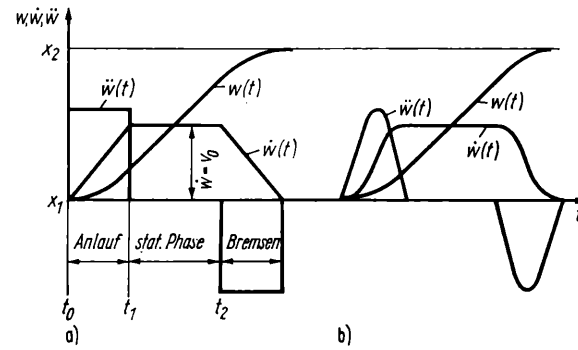


Bild 4.35
Geeignete Führungsgrößenverläufe für Positioniervorgänge

Überführungsfunktionen nach Bild 4.35b sichern ein besonders weiches Anfahren und Abbremsen, da keine Beschleunigungssprünge in der Führungsgrößenvorgabe auftreten. Sie sind besonders für Antriebe hoher Massenbelastung zur Vermeidung unzulässig großer Reaktionskräfte günstig anzuwenden (Reaktionskräfte wirken als Störkräfte auf die übrigen Antriebe der kinematischen Kette.) Eine besonders gute Führung mit einer in der gesamten Bewegungsphase definierten Maximalabweichung ist dann gegeben, wenn

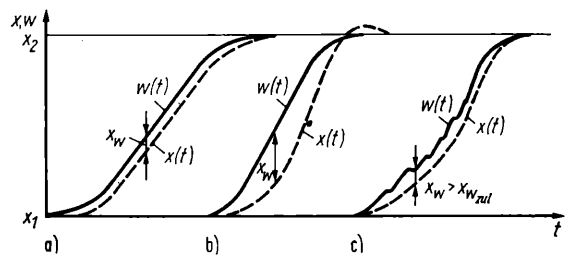


Bild 4.36
Führungsgrößen- und Istwertverläufe
a) der Antrieb kann der Führungsgröße folgen; b) der Antrieb kann der Führungsgröße nicht mehr folgen ($x_w > x_{w\,zul}$); c) Führungsgrößenstop bei $x_w > x_{w\,zul}$

die Regelabweichung ständig überwacht und im Fall der Überschreitung eines vorgegebenen Grenzwerts $x_{w\,zul}$ eine Veränderung der Führungsgröße nicht mehr zugelassen wird ($dw/dt = 0$). Somit stellt sich selbsttätig jene Zeitdehnung im Führungsgrößenverlauf ein, die notwendig ist, damit der Antrieb sich innerhalb eines vorgegebenen Fehlerschlauchs bewegt. Diese Start-Stop-Funktion der Führungsgröße muß mit hoher Frequenz erfolgen, damit keine ruckartigen Bewegungen am Antrieb auftreten (kleine Abtastzeiten). Bild 4.36 zeigt die Wirkungsweise dieses Prinzips während eines Positioniervorgangs von X1 nach X2.

4.4.4. Beispiel einer digitalen Lageregelung

Am Beispiel einer Positionieraufgabe mit elektrohydraulischer Servoregelung soll eine mögliche Realisierungsvariante aufgezeigt werden. Einleitend sei bemerkt, daß es keine geschlossene Theorie oder Projektierungsrichtlinie zur gerätetechnischen Auslegung und zu wählenden Regelkreisstruktur bzw. -dimensionierung für digitale Antriebsregelungen gibt. Nachfolgende Ausführungen können deshalb nur Teilprobleme (und im Rahmen dieses Buches überwiegend nur qualitativ) darlegen.

Der vorliegende Antrieb besteht aus hydraulischem Arbeitszylinder P1-50/22 × 400 TGL 10901 (Kolbenfläche 15,8 cm²), zweistufigem elektrohydraulischem Servoventil TGL 33649 (Nenngröße Q_n = 25 dm³/min, Zeitkonstante druck- und aussteuerungsabhängig $T_{SV} \approx 3 \ldots 4$ ms) und inkrementalem Meßsystem IGR (in Verbindung mit Ritzel–Zahnstange: Meßgliedauflösung 10 μm). Der Betriebsdruck betrage p_0 = 6 MPa. Vom Antrieb ist eine Masse $m \approx 150$ kg horizontal zu bewegen. Es treten Belastungskräfte von $F \approx 2000$ N auf.

Vor Festlegung von Steuerstrategie und Reglerstruktur werden einige Teilprobleme unter besonderer Berücksichtigung der Eigenschaften elektrohydraulischer Antriebe betrachtet.

Eigenschaften des hydraulischen Antriebs

Der elektrohydraulische Antrieb hat eine ausgeprägt nachgiebige statische Kennlinie $\dot{x} = f(F)$. Die Abhängigkeit der Geschwindigkeit $\dot{x}$ von der Belastung F und dem Ansteuerstrom i berechnet sich für Arbeitszylinder mit einem Kolbenflächenverhältnis $\varphi = A_1/A_2 = 1$ zu

$$\dot{x} = \frac{G_{max}}{A}\sqrt{p_0 - \frac{F}{A}}\,\frac{i}{i_{max}} = \frac{Q_n}{A}\sqrt{\frac{p_0}{p^*}\left(1 - \frac{F}{p_0 A}\right)}\,\frac{i}{i_{max}}; \tag{4.95}$$

G turbulenter Leitwert eines Steuerspalts

Q_n Servoventilnenngröße (maximaler Ölstrom bei Kurzschluß der gesteuerten Leitungen und einem Druckabfall Δp = 3,5 MPa über jedem Steuerspalt, d.h. p^* = 7 MPa über dem gesamten Ventil).

Nach Gl. (4.95) besteht bei veränderlicher Belastung keine Proportionalität zwischen Steuerstrom i und Geschwindigkeit $\dot{x}$ (Bild 4.37). Diese Eigenschaft, gepaart mit den Problemen bei der Erfassung niedriger Geschwindigkeiten (vgl. Abschn. 3.4.4. – Stell-

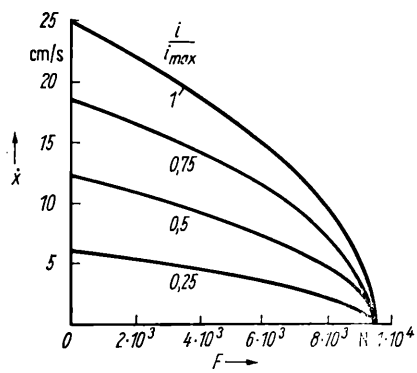


Bild 4.37
Statisches Kennlinienfeld des Servoantriebs (Beispiel)

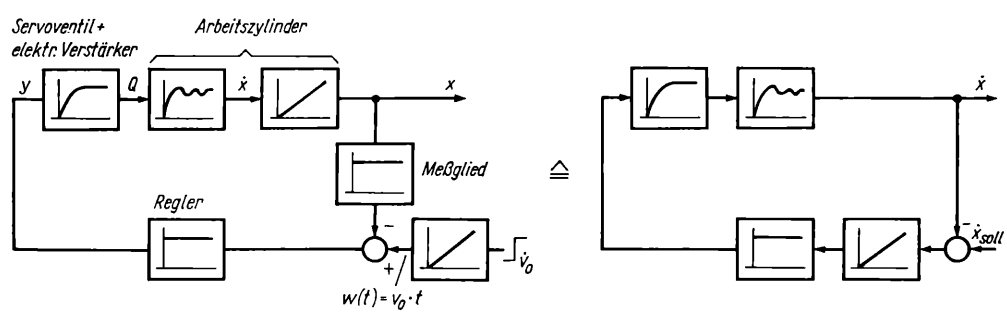
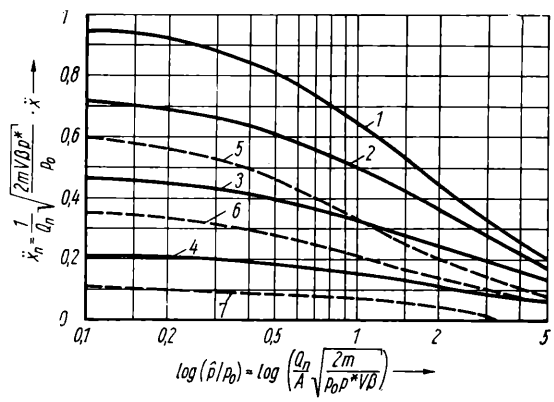


Bild 4.38. Bei Aufschaltung einer stetigen Wegfunktion liegt eine Geschwindigkeitsregelung vor

glied greift i. allg. ohne zusätzliche Getriebe am zu bewegenden Objekt an), macht einen direkten unterlagerten Geschwindigkeitsregelkreis (wie meist bei elektrischen Antrieben) nicht möglich. Zur definierten Führung bei veränderlichen Belastungen ist eine quasistetige Führungsfunktion erforderlich. Bild 4.38 zeigt am Beispiel einer angelegten Rampenfunktion $w(t) = v_0 t$, daß unter diesen Bedingungen eine Geschwindigkeitsregelung (I-Regler) vorliegt, ohne daß dabei die Geschwindigkeit explizit gemessen wird.

Hydraulische Antriebe sind durch eine hohe Energiedichte charakterisiert. Obwohl der Druckaufbau in den Zylinderräumen (und damit die Kraftentwicklung) durch Ölkompressibilität und Drosselung in den Steuerspalten des Servoventils verzögert erfolgt (RC-Glieder), werden beträchtliche Beschleunigungswerte erreicht. Dagegen nimmt das Beschleunigungsvermögen mit wachsender Antriebsgeschwindigkeit erheblich ab. (Struktur entspricht etwa einem E-Motor mit Drehzahlstellung mittels Vorwiderstand.)



Nr.	$\ddot{x}_n = \frac{A}{Q_n}\sqrt{\frac{p^*}{p_0}} \cdot \ddot{x}$	$F_n = \frac{F}{p_0 \cdot A}$
1	0	0
2	0,25	0
3	0,5	0
4	0,75	0
5	0	0,6
6	0,25	0,6
7	0,5	0,6

Bild 4.39. Normierte Maximalbeschleunigung eines Servoantriebs mit einem Verbraucher $\varphi = 1$

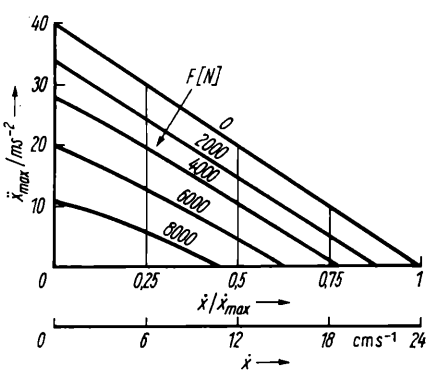


Bild 4.40
Maximalbeschleunigung $\ddot{x} = f(\dot{x})$, Parameter F für betrachtetes Beispiel

Für Antriebe mit $\varphi = 1$ sind unter Vernachlässigung der Servoventilzeitkonstante T_{SV} im Bild 4.39 normierte Grenzbeschleunigungen $\ddot{x}_{max}$ in Abhängigkeit von Geschwindigkeit $\dot{x}$ und Belastung F dargestellt (Analogrechnerergebnisse mit nichtlinearem Modell). Die Originalgrößen für vorliegende Regelstrecke lassen sich daraus ermitteln (Bild 4.40). Bei Anlauf ($\dot{x} = 0$) und Gegenkräften von 2000 N wird $\ddot{x}_{max} \approx 34$ m/s^2 erreicht. Das entspricht bei $m = 150$ kg Reaktionskräften von etwa 5000 N auf das Antriebsgestell. Bei 70% der Maximalgeschwindigkeit und 50% der möglichen Maximalkraft hingegen ist der Antrieb überhaupt nicht mehr beschleunigungsfähig. Dieser Sachverhalt ist bei der Vorgabe der Führungsgrößen unbedingt zu berücksichtigen.

Modell für Regelstrecke

Unter Berücksichtigung der Ölkompressibilität (sie ist aus praktischen Erfahrungen für eine wirklichkeitsnahe Modellierung kaum zu vernachlässigen) gilt in Näherung das linearisierte Modell gemäß Abschn. 4.2.4. Das Servoventil ist als PT_1-Glied hinreichend genau zu beschreiben:

$$\frac{x(p)}{y(p)} = \frac{K_{SV}}{1 + T_{SV}p} \frac{C/A}{1 + 2DTp + T^2p^2} \frac{1}{p}. \tag{4.96}$$

Wesentlich ist, zu wissen, daß sich die Zeitkonstante T über dem Kolbenhub verändert und die Dämpfung D in Abhängigkeit von der Ventilaussteuerung um Größenordnungen schwanken kann.

Die maximale Zeitkonstante T (konstante Masse vorausgesetzt) tritt bei $\varphi = 1$ in Kolbenmittelstellung auf:

$$T = \frac{1}{A}\sqrt{\frac{mV\beta}{2}}; \tag{4.97}$$

β Kompressibilitätsfaktor des Öles etwa $8 \ldots 10 \cdot 10^{-4}$ MPa^{-1}.

Die Gesamtdämpfung D

$$D = D_m + D_h = \frac{K_D}{2A}\sqrt{\frac{V\beta}{2m}} + \frac{\varphi_E + G_L}{2A}\sqrt{\frac{2m}{V\beta}}; \tag{4.98}$$

K_D Faktor der geschwindigkeitsproportionalen Reibung

besteht aus zwei Anteilen. Während der mechanische Anteil D_m annähernd konstant bleibt, verändert sich der hydraulische Anteil D_h stark mit der Aussteuerung (starke Verminderung der Druckverstärkung E mit wachsender Aussteuerung). In der Position $\dot{x} \to 0$ sind Gesamtdämpfungswerte $D = 0{,}05 \ldots 0{,}1$ keine Seltenheit. Für Stabilitätsberechnungen und Reglerdimensionierung sind T_{max} und D_{min} innerhalb des Arbeitsbereichs zugrunde zu legen.

Reglerstruktur

Der Hydraulikantrieb greift wegen seiner großen Stellkräfte i.allg. unmittelbar ohne zwischengeschaltetes Getriebe am zu bewegenden Objekt an. Das hat zur Folge, daß Änderungen von Massenbelastungen am Roboter direkt in die Zeitkonstanten der Strecke eingehen, während bei Elektroantrieben eine Masseänderung mit dem Faktor $1/\ddot{u}^2$ am Motor wirksam wird ($\ddot{u}$ Getriebeübersetzungsverhältnis). Damit ist der Variationsbereich der Regelstreckenparameter u. U. im Arbeitsbereich relativ groß (ausgeprägt bei Robotern in Gelenkbauweise). Wird mit festen Reglereinstellwerten gearbeitet (das ist z. Z. die überwiegende Ausführung), dann sind zur Sicherung der Stabilität die Reglerparameter dem ungünstigsten Arbeitspunkt anzupassen. Damit wird in anderen Arbeitspunkten auf eine dort vorhandene Reserve an dynamischer und statischer Güte verzichtet. Die Möglichkeit einer Parameterverstellung (z. B. von der Information „Werkstück gegriffen" oder „Extremstellung anderer Antriebe") sollte genutzt werden, wobei eine sprungförmige Verstellung nicht in der Bewegungsphase vorzunehmen ist (ruckartige Bewegungen). Für eine exakte Positionierung ist ein integraler Anteil im Regler unumgänglich. Ein durchgängig wirksamer I-Anteil führt zu einem strukturbestimmten Überschwingen und gestattet kein überschwingfreies Einlaufen in die Position. Es ist deshalb nur in der Umgebung der Position zuzuschalten bzw. in der Bewegungsphase zu begrenzen. Im Zu-

sammenhang mit der Ölkompressibilität und der Reibungscharakteristik zu bewegender Objekte (Bild 4.41) hat der I-Anteil eine weitere ungünstige Eigenschaft. Liegt eine Positionsabweichung vor, wird die Regelabweichung so lange aufintegriert, bis die Stellgröße ausreicht, die Haftreibung zu überwinden. Bei einsetzender Bewegung entspannt sich die Ölsäule, und die degressive Reibkennlinie (geringere Gegenkräfte bei $\dot{x} > 0$) läßt den Antrieb über die Position hinausschwingen. Der Vorgang vollzieht sich in Gegenrichtung in gleicher Weise, was zum ständigen Schwingen mit niedriger Frequenz um die Position führt, falls der Antrieb nicht „zufällig" in der Reibungshysterese zum Stehen

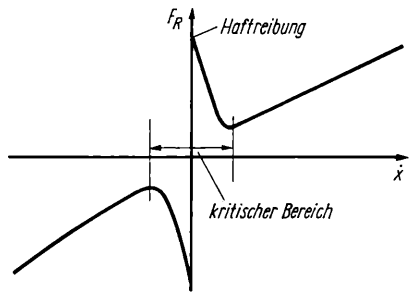


Bild 4.41
Typische Reibcharakteristik $F_R = f(\dot{x})$ von Verbrauchern

kommt. Häufig hilft nur eine Unempfindlichkeitszone im I-Anteil, innerhalb der die I-Komponente abgeschaltet wird, was zwangsläufig zu einem höheren Positionierfehler führt. (Die erreichbare Genauigkeit wird nicht mehr durch die Meßgliedauflösung bestimmt.) D-Anteile im Regler bewirken eine Stabilisierung des Regelkreises bzw. eine stärkere Bewertung von Führungsgrößenänderungen. Wegen der kleinen Dämpfungswerte elektrohydraulischer Antriebe steht der erstgenannte Aspekt meist im Vordergrund, so daß es häufig günstiger ist, nur die Regelgröße anstelle der gesamten Regelabweichung differentiell zurückzuführen. Da die direkte Messung der Istgeschwindigkeit am Hydraulikantrieb nicht möglich ist ($\dot{x} \triangleq dx/dt$), ist die D-Rückführung durch numerische Differentiation – bewertete Differenz $(x_k - x_{k-1})/t_{ab}$ – zu bilden. Zur Einhaltung der für einen wirksamen D-Anteil notwendigen Phasenbeziehungen ist eine kleine Abtastzeit t_{ab} erforderlich. Bei Antrieben kleiner Zeitkonstanten bedeutet dies, daß (abhängig von der Meßgliedauflösung) zwischen zwei Tastzyklen nur kleine Istwertdifferenzen auftreten. Diese müssen, um den D-Anteil zur Wirkung zu bringen, mit einem großen Faktor multipliziert werden, was zwangsläufig zu großen Quantisierungseinheiten der Stellgröße führt. Dadurch kann u. U. sogar ein gegenteiliger Effekt, also eine Destabilisierung des Antriebs, eintreten. So konnte z. B. bei einem Antrieb (dominierende Zeitkonstante etwa 6 ms, Meßgliedauflösung 5 μm) für Abtastzeiten $t_{ab} = 0{,}3$ bis 10 ms keine Einstellung gefunden werden, die einen positiven Einfluß der D-Komponente erkennen ließ.

Abtastfrequenz

Prinzipiell gilt, daß ein Abtastsystem nur für den Grenzfall $t_{ab} \to 0$ die Qualitätsparameter eines entsprechenden analogen Regelkreises erreicht. Jedoch ist nachweisbar, daß ab einer bestimmten Grenze eine weitere Verringerung der Abtastzeit nur unwesentliche Verbesserungen zu erzielen sind. In [4.16] wird der Einfluß der Abtastzeit auf Stabilität, Bandbreite und Steuerbarkeit des Regelkreises abgeleitet. An dieser Stelle soll nur der Stabilitätseinfluß betrachtet werden. Im Bild 4.42 ist die Abhängigkeit der kritischen Kreisverstärkung (integraler Übertragungsfaktor des offenen Kreises) für vorliegende Regelstrecke mit P-Regler dargestellt (Grenzfall synchroner Abtastung). Die Berechnung

erfolgte über die Beschreibungsfunktion mit dem linearisierten Modell nach (Gl. 4.96) [4.15]. Die Maximalwerte der Kreisverstärkung werden, nahezu unabhängig von der Dämpfung, bei $\omega_{ab}T \approx 2{,}5$ erreicht. Mit den Werten der Regelstrecke

Kolbenfläche $A = 15{,}8\ \text{cm}^2$; Masse $m = 150\ \text{kg}$; Volumen zwischen Kolben und Servoventil $V = 500\ \text{cm}^3$; Kompressibilitätsfaktor $\beta = 8 \cdot 10^{-4}\ \text{MPa}^{-1}$

ergibt sich nach Gl. (4.97) $T \approx 4$ ms, woraus sich eine Abtastfrequenz von etwa 100 Hz (Abtastzeit $t_{ab} = 10$ ms) berechnet. An dieser Stelle des Diagramms (Bild 4.42) ist $K_{I\,krit}\,T \approx 0{,}48$ ablesbar, woraus der Originalwert

$$K_{I\,krit} = K_{SV}\frac{C}{A}K_M K_R = \frac{0{,}48}{T} = 120\ \text{s}^{-1}$$

zu bestimmen ist. Der ermittelte Wert kann als grober Richtwert angesehen werden (Vernachlässigungen im Modellansatz, systemeingeschränkte Gültigkeit der Beschreibungsfunktion). Die Einbeziehung coulombscher Reibung bewirkte im vorliegenden Beispiel eine Veränderung der kritischen Verstärkungswerte bis 30%, wobei diese bei hohen Tastfrequenzen wie eine Dämpfungserhöhung wirkte, während bei niedrigen Tastfrequenzen ein destabilisierender Effekt auftrat.

Ausführungsvariante

Bild 4.43 zeigt schematisch die gewählte Regelkreisstruktur mit Führungsgrößengenerator zur Vorgabe einer Sollwertfunktion mit sinoidischem Anfahr- und Abbremsvorgang (vgl. Bild 4.35b). Die angestrebte stationäre Verfahrgeschwindigkeit sei $\dot{x}_{soll} = v_0$, die Maximalbeschleunigung $\ddot{x}_{max} = b_0$ vorgegeben. Damit lauten die Gleichungen der zu generierenden Führungsfunktion $w(t)$:

Anlauf

$$\left.\begin{aligned}
\ddot{w}(t) &= b_0 \sin \omega t = b_0 \sin \frac{2b_0}{v_0}\,t \\
\dot{w}(t) &= \frac{v_0}{2}(1 - \cos \omega t) = \frac{v_0}{2}\left(1 - \cos \frac{2b_0}{v_0}\,t\right) \\
w(t) &= w(t_0) + \frac{v_0}{2}\left(t + \frac{1}{\omega}\sin \omega t\right) = w(t_0) + \frac{v_0}{2}\left(t + \frac{v_0}{2b_0}\sin \frac{2b_0}{v_0}\,t\right)
\end{aligned}\right\} \tag{4.99}$$

stationäre Phase

$$\left.\begin{aligned}
\ddot{w}(t) &= 0 \\
\dot{w}(t) &= v_0 \\
w(t) &= w(t_1) + v_0(t)
\end{aligned}\right\} \tag{4.100}$$

Abbremsen

$$\left.\begin{aligned}
\ddot{w}(t) &= -b_0 \sin \omega t = -b_0 \sin \frac{2b_0}{v_0}\,t \\
\dot{w}(t) &= \frac{v_0}{2}\left(1 + \cos \omega t\right) = \frac{v_0}{2}\left(1 + \cos \frac{2b_0}{v_0}\,t\right) \\
w(t) &= w(t)_2 + \frac{v_0}{2}\left(t - \frac{1}{\omega}\sin \omega t\right) = w(t_2) + \frac{v_0}{2}\left(t - \frac{v_0}{2b_0}\sin \frac{2b_0}{v_0}\,t\right).
\end{aligned}\right\} \tag{4.101}$$

Maximalbeschleunigung b_0 und -geschwindigkeit v_0 sind in Übereinstimmung mit der Leistungsfähigkeit des Antriebs (Bild 4.40) festzulegen. Die Wegsollwerte $w_i(t)$ können für die diskreten Zeiten nt_{ab} berechnet und als Tabelle abgespeichert werden. Der Übergang von Gl. (4.99) auf Gl. (4.100) ist zwar eindeutig nach n Tastperioden mit

$$n \geqq \frac{\pi v_0}{2 b_0 t_{ab}} \tag{4.102}$$

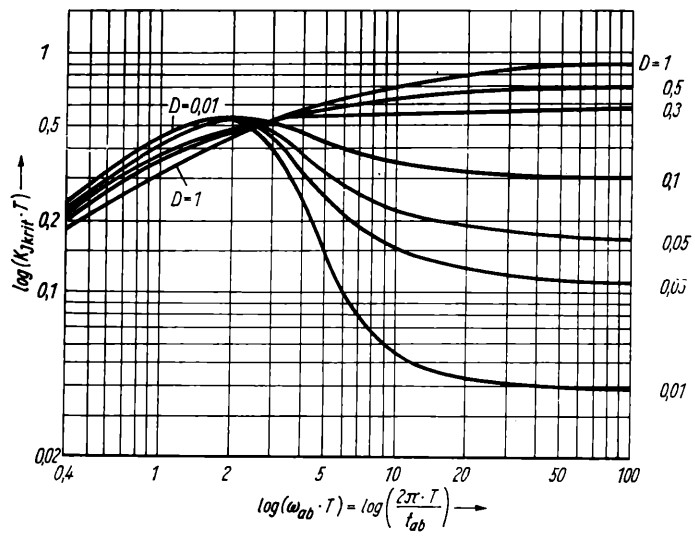


Bild 4.42
Abhängigkeit der Stabilitätsgrenze von der Abtastzeit für ein Antriebsmodell nach Gl. (4.96)
T_{SV}/T = 1, P-Regler, synchrone Abtastung

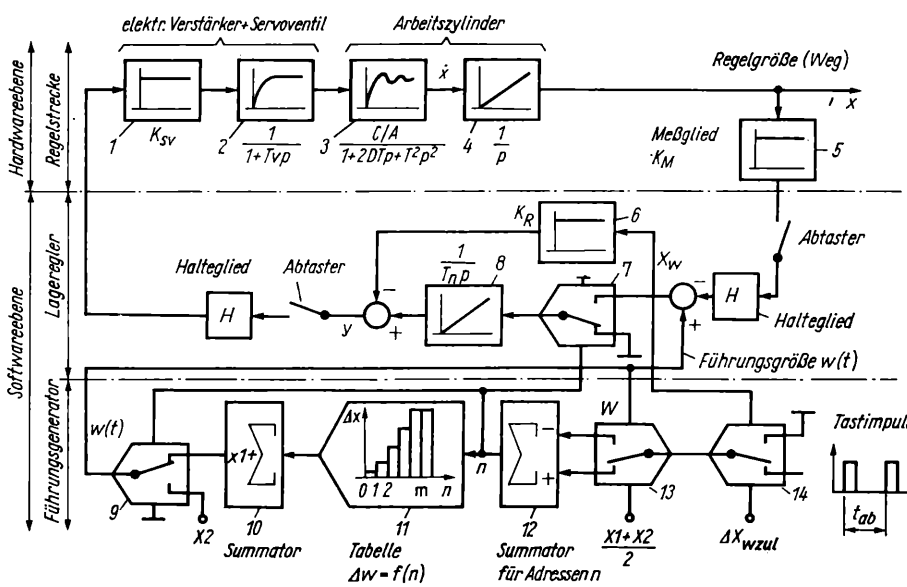


Bild 4.43. Signalflußbild eines Positionierantriebs mit Führungsgrößengenerator

festgelegt, jedoch der Übergang von Gl. (4.100) auf Gl. (4.101) (Beginn des Abbremsens) ist unklar. Deshalb wird zweckmäßigerweise der je Tastintervall erforderliche Wegzuwachs

$$\Delta w_i = w(nt_{ab}) - w((n-1)\,t_{ab}) \tag{4.103}$$

extern berechnet und abgespeichert. Diese Tabelle wird beim Anfahren aufwärts und nach Überschreiten der halben Positionsdifferenz $|\Delta x| = |(X1 - X2)/2|$ abwärts abgearbeitet, wobei im Rechner der jeweils aktuelle Führungsgrößenwert $w(nt_{ab})$

$$w(nt_{ab}) = X1 + \sum_{i=1}^{n} \Delta w_i \tag{4.104}$$

berechnet wird. Im Bild 4.43 stellt das Glied *11* diese Zuwachstabelle dar. Beim Startsignal zum Verfahren von X1 nach X2 stehen die Komparatoren *13* und *14* in der gezeichneten Stellung. Je Tastintervall (symbolisiert durch die Impulsfolge am Glied *14*) wird der Adressenzähler *12* für die Wegzuwachstabelle *11* erhöht und die Zuwachsinkremente im Summator *10* zum aktuellen Führungsgrößenwert aufaddiert. Bei Überschreitung des halben Verfahrwegs schaltet der Komparator *13* μm; der Adressenzähler *12* zählt abwärts, und Tabelle *11* wird bis zum Wert 0 abgearbeitet. Da i.allg.

$$|\Delta x| = |X1 - X2| \neq 2 \sum_{i=1}^{n} \Delta w_i$$

ist, wird nach Abarbeiten der Tabelle über Komparator *9* die Zielposition X2 als letzter Lagesollwert am Regelkreiseingang wirksam. Geregelt wird in der Verfahrphase mit P-Regler (Glied *6*), während in der Umgebung der Zielposition X2 ein I-Anteil (Glied *8*) zugeschaltet wird. Sollte der Antrieb der berechneten Führungsgröße nicht mehr mit ausreichender Genauigkeit folgen können ($x_w > x_{wzul}$), schaltet Komparator *14*, wodurch ein weiterer Führungsgrößenzuwachs so lange nicht mehr erfolgt, bis x_w wieder in den zulässigen Toleranzbereich eingelaufen ist.

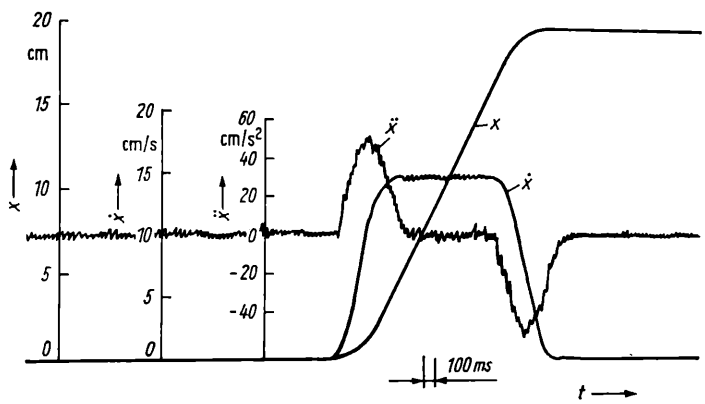


Bild 4.44
Oszillografierte Übergangsfunktionen

Diese Struktur, die im Detail weitere Nuancen zuläßt, erfordert für ihre Implementierung auf einem Mikrorechner K1520 etwa 1,2 kByte Speicherkapazität. Die Programmabarbeitungszeit beträgt etwa 0,8 ms. Bild 4.44 zeigt am praktischen System oszillografierte Übergangsfunktionen.

4.5. Adaptive Regelung

Ändern sich die Parameter der Regelstrecke in nicht vorhersehbarer Weise, sind zur Erreichung vorgegebener Gütekriterien adaptive Regelverfahren notwendig. Die Theorie der adaptiven Regelung ist erst in jüngster Zeit durch grundlegende Arbeiten von Zypkin [4.17], Bellmann [4.18], Astroem [4.19] und Feldbaum [4.20] soweit entwickelt

worden, daß praktische Einsatzfälle realisiert werden können; das betrifft insbesondere Stabilitätsprobleme und die erforderlichen Verfahren zur Identifikation und Parameterschätzung.

Adaptive Regelverfahren lassen sich grob in zwei Gruppen einteilen:

Die erste Methode beruht auf der Nutzung eines Referenzmodells und wird als MRAC (*m*odel *r*eference *a*daptive *c*ontrol) bezeichnet. Bild 4.45 zeigt schematisch das Prinzip. Die Eingangsgröße wird einem parallel zur Regelstrecke liegenden Modell zugeführt; dieses Referenzmodell enthält den determinierten Teil der Regelstrecke. Die durch Parameteränderungen hervorgerufenen Abweichungen der zu regelnden Prozeßwerte werden durch Vergleich mit dem Ausgangssignal des Referenzmodells ermittelt und steuern die Einstellung der Regelparameter so, daß die Abweichungen zwischen Modell- und Prozeßausgang eliminiert werden. Auf die unterschiedlichen Methoden zur Ermittlung der Regelparameter soll hier nicht eingegangen werden; sie sind in [4.21] ausführlich dargestellt. Die Anwendung dieses Verfahrens zur Lageregelung der Roboterachsen beinhaltet somit die Nutzung des dynamischen Modells, das bis auf stochastische Reibungsanteile und im Greifer befindliche Massen oder am Effektor angreifende äußere Kräfte grundsätzlich bekannt ist. Um den zur Implementierung des Modells erforderlichen Aufwand zu reduzieren, ist es aber auch denkbar, das Referenzmodell abzurüsten, z.B. durch Vernachlässigung der geschwindigkeitsabhängigen Terme in Gl. (4.5), und nur die zustandsabhängigen Anteile einzubeziehen. Die vernachlässigten Terme werden dann als äußere Störungen aufgefaßt. Dies bedeutet aber erhöhte Anforderungen an das Zeitverhalten der Parameterbestimmung.

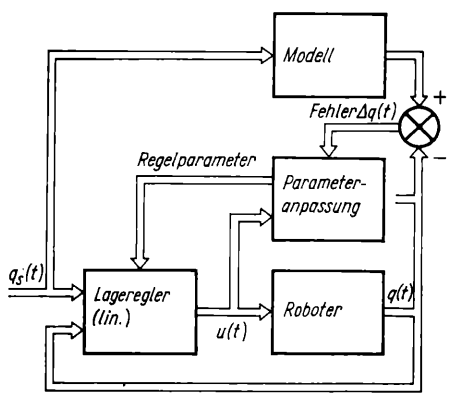


Bild 4.45. Adaptive Regelung mit Referenzmodell

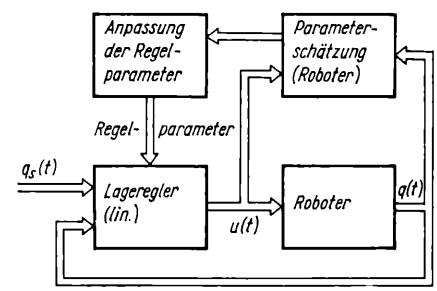


Bild 4.46. Selbsteinstellendes Regelungssystem

Eine konsequente Weiterführung der Vernachlässigungen führt zur zweiten Methode der adaptiven Regelung, den selbsteinstellenden Verfahren STUR (*s*elf *tu*ning *r*egulation). Das Prinzip ist im Bild 4.46 dargestellt. Durch Parameterschätzung werden aus den Steuersignalen $u(t)$ und der Regelgröße Parameter der Regelstrecke ermittelt, die das Verhalten eines linearen Reglers modifizieren. Wendet man diese Methode auf die Lageregelung der Roboterachsen an, müssen insbesondere zeitliche Forderungen an die Parameterschätzung erfüllt werden. Die Änderungsgeschwindigkeit der dynamischen Parameter des Manipulators bestimmt das notwendige Konvergenzverhalten der Schätzalgorithmen.

Mit wenig Aufwand realisierbare Verfahren sind möglich, wenn vereinfachende Voraussetzungen über die Änderungen der Regelstrecke und die Möglichkeiten der Identi-

fikation getroffen werden können. Hierzu sollen nachfolgend zwei Beispiele diskutiert werden, bei denen eine Adaption bezüglich der durch im Greifer befindliche Massen hervorgerufenen Änderungen der Trägheitsmomente erfolgt.

Der Einfluß des Trägheitsmoments auf die Gesamtübertragungsfunktion des Regelkreises mit unterlagerter Geschwindigkeitsregelung wird mit den im Abschnitt 4.3.2. angenommenen P-Reglern gemäß (4.83) bei Vernachlässigung der Reibung zu

$$G(p) = \frac{X(p)}{W(p)} = \frac{K_m K_x}{p^2 J_g + p K_v K_m + K_x K_m}; \tag{4.105}$$

J_g Gesamtträgheitsmoment der Achse.

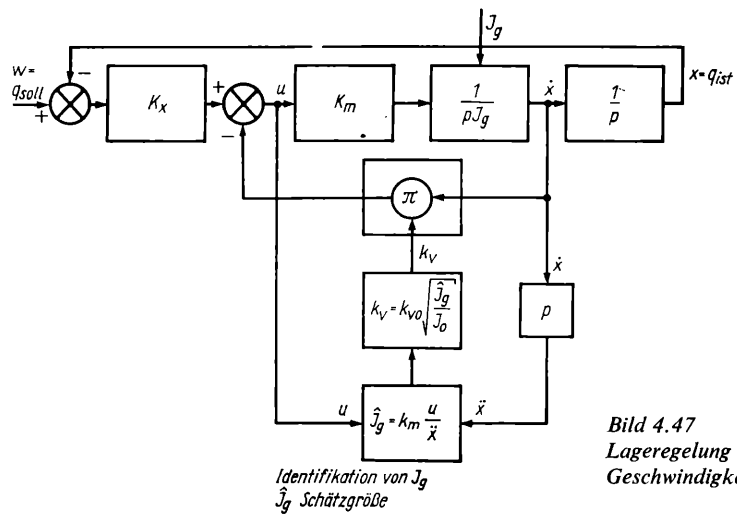


Bild 4.47
Lageregelung mit unterlagerter Geschwindigkeitsregelung

Die Struktur wird im Bild 4.47 gezeigt. Ist J_M das Eigenträgheitsmoment des Antriebs, läßt sich dies auf die Achse umrechnen mittels

$$J_g = J + J_M a^2;$$

a Getriebeübersetzungsfaktor.

Die Variation der Trägheitsmomente führt zu einer Variation der mechanischen Resonanzfrequenz des Systems. Es gilt

$$\omega_m = \omega_0 \sqrt{\frac{J_0}{J_g}}, \tag{4.106}$$

wenn ω_0 die Resonanzfrequenz beim Nennträgheitsmoment J_0 ist.

Gl. (4.105) stellt die Übertragungsfunktion eines Systems 2. Ordnung

$$G(p) = \frac{1}{p^2 + 2\beta\omega p + \omega^2} \tag{4.107}$$

mit der charakteristischen Frequenz ω und dem Dämpfungsfaktor β dar.

$$\omega = \sqrt{\frac{K_x K_m}{J_g}}, \qquad \beta = \frac{K_v K_m}{2\sqrt{J_g K_x K_m}}. \tag{4.108}$$

Zur Vermeidung mechanischer Schwingungen muß die Regelung so ausgelegt werden, daß $\omega \approx 0{,}5\,\omega_m$. Die kürzesten Übergangszeiten werden für $\beta = 1$ erreicht. Damit wird

$$K_v K_m = 2\sqrt{J_g K_x K_m}. \tag{4.109}$$

Um sowohl Überschwingen der Positionsendwerte als auch mechanische Schwingungen zu vermeiden, muß also gelten

$$0{,}5\,\omega_0 \sqrt{\frac{J_0}{J_g}} > \sqrt{\frac{K_x K_m}{J_g}},$$

d.h., der maximale Wert von $K_x K_m$ wird

$$(K_x K_m)_{max} = \tfrac{1}{4}\omega_0^2 J_0.$$

Bei der praktischen Auslegung von Lageregelkreisen wird diese Bedingung oft nicht eingehalten, um kürzere Positionierzeiten zu erreichen. Ein geringes Überschwingen wird dabei bewußt in Kauf genommen.

Äquivalente Beziehungen gelten für die Massenabhängigkeit bei Schubachsen.

Aus Gl. (4.109) läßt sich die kritische Dämpfung für das Trägheitsmoment J_g durch entsprechende Wahl des Verstärkungsfaktors für die Geschwindigkeitsrückkopplung bestimmen. Geschieht dies für das maximal auftretende Trägheitsmoment, wird zwar das Überschwingen vermieden; die Positionierzeiten bei geringerer Belastung werden jedoch vergrößert. Optimale Positionierzeiten werden erreicht, wenn K_v dem momentanen Trägheitsmoment entsprechend berechnet wird. Dabei kann die Rückkopplungsverstärkung gemäß

$$K_v = K_{v0}\sqrt{\frac{J_g}{J_0}} \tag{4.110}$$

aus der Lastvariation bestimmt werden. Das kann wie im Bild 4.47 durch Auswertung der Stellgröße u und der Beschleunigung $\ddot{x}$ erfolgen. Aus der zugrunde liegenden Bewegungsgleichung

$$J_g \ddot{x} = K_m u \tag{4.111}$$

läßt sich durch Umstellen nach J_g ein Schätzwert $\hat{J}_g$ ermitteln; damit kann K_v gemäß (4.110) ständig nachgestellt werden. Offenbar ist dieses Verfahren nur anwendbar, wenn der Drehzahlregelkreis im linearen Bereich arbeitet.

Mit dem dargestellten Verfahren ist es prinzipiell auch möglich, mehrere gekoppelte Achsen unabhängig voneinander adaptiv zu regeln, d.h., die Parameter K_{vi} an die zu-

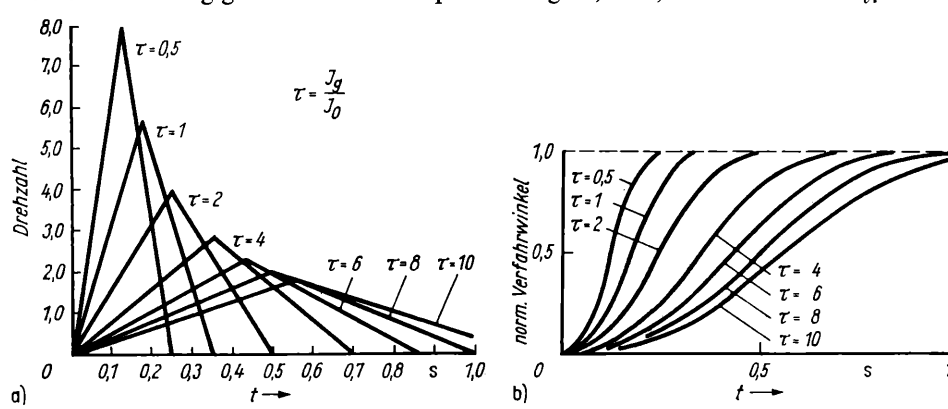


Bild 4.48. Drehzahl und Verfahrwinkel bei zeitoptimalem Regler mit der Lastvariation τ als Parameter

standsabhängigen Trägheitsmomente und die auftretenden Koppelkräfte anzupassen. Hierbei können jedoch Stabilitätsprobleme auftauchen.

Kommt gemäß Abschnitt 4.4.1. ein zeitoptimaler Regler zur Anwendung, besteht das Problem der Adaption darin, bei variabler Last die Umschaltpunkte zwischen Beschleunigen und Abbremsen zu bestimmen. Bild 4.48 zeigt den zeitlichen Verlauf der Drehzahl und des Verfahrwinkels bei der Lastvariation τ.

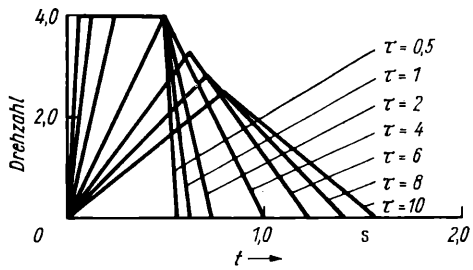


Bild 4.49. Drehzahlverlauf bei beschränkter Ableitung der Steuergröße (nach [4.8])

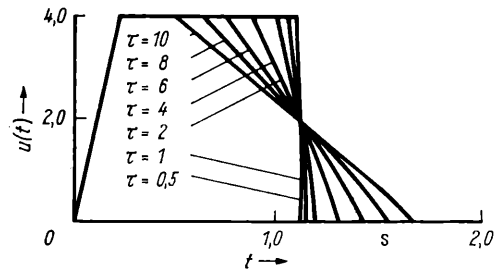


Bild 4.50. Führungsgröße zur angenähert zeitoptimalen Regelung (nach [4.8])

Bei beschränkter erster Ableitung der Steuergröße ergibt sich der im Bild 4.49 dargestellte Verlauf der Winkelgeschwindigkeit. In diesem Fall ist ein angenähert zeitoptimales Verfahren möglich, wenn gemäß Bild 4.50 eine Führungsgröße erzeugt wird, die das rechtzeitige Einsetzen der Bremsphase sichert. Ein entsprechender Algorithmus wird in [4.8] beschrieben.

Beim zeitoptimalen Positionieren wird davon ausgegangen, daß sich die Trägheitsmomente während des Verfahrens nicht ändern. Die durch Verkopplungen auftretenden Variationen können damit nicht berücksichtigt werden. Es kann nur eine Adaption an die durch äußere Belastung des Manipulators bedingten Parameteränderungen erfolgen. Zur Identifikation der Belastung sind unterschiedliche Verfahren geeignet. Die direkte Messung durch Kraft-Moment-Sensoren ist im Abschnitt 7. dargestellt. Eine einfache Identifikation aus der Regelabweichung ist möglich, wenn sie innerhalb einer Phase konstanter Beschleunigung erfolgt.

4.6. Bahnsteuerung

Die Steuerung des Endeffektors (Werkzeug, Greifer) auf einer vorgegebenen Raumkurve erfordert im allgemeinen Fall die Koordinierung der einzelnen Bewegungsachsen und wird mithin eine Steuerung mit Bahnverhalten.

Nach der Art der Führungsgrößenerzeugung lassen sich solche Steuerungen unterteilen in

- Multipunktsteuerungen, die die Führungsgrößen als Sollwertfolge vollständig in einem Programmierlauf während der Bewegung des Grundgeräts oder eines Hilfsgeräts aufnehmen und abspeichern;
- Bahnsteuerungen, die die Führungsgrößen durch Berechnung einer Raumkurve erzeugen, deren charakteristische Werte in die Steuerung eingegeben und abgespeichert werden.

Die Führungsgrößen werden dabei in beiden Fällen so schnell an die Lageregelkreise ausgegeben, daß aufgrund der dynamischen Eigenschaften der Antriebe und der an-

getriebenen Achse ein kontinuierlicher Bewegungsverlauf erreicht wird (Folgeregelung). Bahnsteuerungen sind Voraussetzung für den Einsatz von Industrierobotern beim Farbspritzen, Lichtbogenschweißen, Entgraten und ähnlichen technologischen Aufgaben.

Haupteinsatzgebiet für Multipunktsteuerungen sind hydraulisch angetriebene Roboter, die durch Freischalten der Antriebe leicht bewegt und Sollwerte von Bahnkurven im direkten Teach-in erstellt werden können.

Das Erzeugen einer Sollwertfolge im indirekten Teach-in erfordert die Ansteuerung aller Raumpunkte mit der für die Folgeregelung notwendigen Auflösung und ist sehr zeitaufwendig.

Reduziert man die Anzahl der Bahnstützpunkte, muß in geeigneter Weise interpoliert werden. Die Interpolation kann dabei Zwischenwerte der Sollwertfolge einzelner Achsen aus den gespeicherten Positionswerten dieser Achse oder Zwischenwerte einer in natürlichen Koordinaten vorliegenden Raumkurve aus deren analytischer Darstellung erzeugen.

Im zweiten Fall ist zur Erzeugung einer Sollwertfolge für die Lageregelkreise die Umrechnung der in natürlichen Koordinaten vorliegenden Interpolationswerte in das achsspezifische Koordinatensystem erforderlich. Die Lösung dieser Koordinatentransformation – auch als inverses kinematisches Problem bekannt – ist für Industrieroboter mit einer Anzahl von Freiheitsgraden $n > 3$, insbesondere bei rotatorischen Achsen, mit hohem rechentechnischem Aufwand verbunden.

Die Koordinatentransformation bildet gleichzeitig eine notwendige Voraussetzung für die Steuerung nach Sensordaten, die in natürlichen Koordinaten erfaßt werden.

Handhabungsaufgaben, bei denen ein Werkzeug auf einer durch den technologischen Bearbeitungsprozeß vorgegebenen Bahn geführt wird, erfordern neben der Positionierung auch eine Orientierung des Werkzeugs. Eine durch das Werkzeug gelegte Gerade ist in einem durch den Bearbeitungsvorgang gegebenen funktionalen Zusammenhang mit der Positionierung zu führen.

Die allgemeine Lösung dieser Aufgabe erfordert sechs Freiheitsgrade des mechanischen Systems, da Positionierung und Orientierung durch sechs voneinander unabhängige Variable beschrieben werden. Mit Einschränkungen für die Orientierung sind auch Systeme mit fünf Freiheitsgraden einsetzbar.

Bahnsteuerungen werden nicht nur aus technologischen Forderungen eingesetzt; sie verbessern auch die Einsatzparameter für Punkt-zu-Punkt-Bewegungen. Insbesondere sind zu nennen:

- Verkürzung der Positionierzeiten durch gleichzeitige Bewegung aller Achsen,
- Verringerung der am gegriffenen Werkstück auftretenden Kräfte während der Bewegung durch kontinuierliche Geschwindigkeitsverläufe,
- Einsparung des spezifischen Energieverbrauchs durch eine geringere Anzahl von Beschleunigungsphasen.

Bahnsteuerungen erfordern neben einem höheren rechentechnischen Aufwand eine größere Leistungsreserve bei der Auslegung der Primärenergiewandlung (Hydraulikpumpe, Netzteile) für gleichzeitige schnelle Bewegungen mehrerer Antriebe.

4.6.1. Bahnsteuerverfahren bei Multipunktsteuerungen

Multipunktsteuerungen werden im direkten Teach-in programmiert. Dabei gelangen in der Programmierungsphase (Teach-in) die im zeitlichen Rhythmus abgetasteten und in Digitalgrößen umgewandelten Achspositionen in der Reihenfolge ihres zeitlichen Ein-

treffens in den Speicher der Steuerung. In der Arbeitsphase (Playback) werden diese Positionswerte in der gleichen Reihenfolge aus dem Arbeitsspeicher ausgelesen und den Servoreglern als Sollwerte zugeführt. Dabei entstehen während der Programmierphase im Prinzip Wertetabellen für die Positionszeitverläufe aller Achsen im Arbeitsspeicher, wobei die Zeitvariable in der Regel durch die ständig weiterzählende Adresse repräsentiert wird (Bild 4.51).

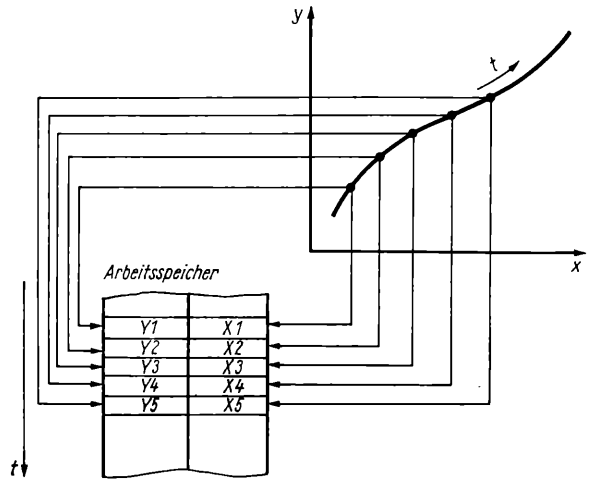


Bild 4.51
Prinzip der Bahnkurvenabspeicherung und -erzeugung bei einer Multipunktsteuerung

Bei Multipunktsteuerungen bedingen die Forderungen nach Präzision und Qualität der Bewegungen zusammen mit den Zeitkonstanten der Servoregler relativ hohe Taktfrequenzen der Sollwertvorgaben (typisch 80 Hz), was bei der direkten Speicherung aller Positionswerte zu einem beträchtlichen Speicherplatzbedarf für technologisch relevante Programmlaufzeiten führt. Bild 4.52 zeigt dazu die Abhängigkeit des Speicherplatzbedarfs von der Abtastfrequenz und der technologischen Programmlaufzeit für einen

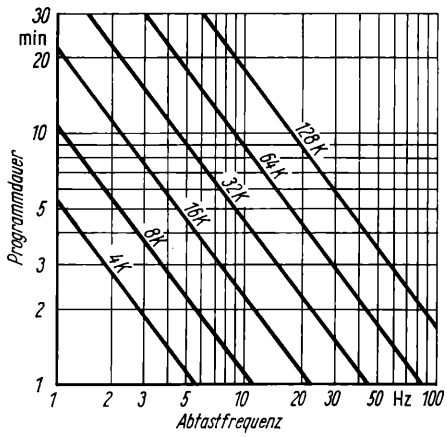


Bild 4.52
Speicherplatzbedarf bei einer einfachen Multipunktsteuerung in Abhängigkeit von der Abtastfrequenz und der Programmdauer (6achsiger Industrieroboter, Speicherwortbreite 2 byte je Achsposition)

6achsigen Roboter, wobei je Positionswert eine Speicherwortbreite von 2 byte vorgesehen ist. Man erkennt, daß bei einer Abtastfrequenz von beispielsweise 50 Hz bereits einige Minuten Programmlaufzeit Speicher von 100 kbyte und mehr erfordern. Deshalb finden in einigen Multipunktsteuerungen Massenspeicher mit seriellem Zugriff (Magnetband, Floppy-Disc [4.22] [4.24]) als Arbeitsspeicher Verwendung. Jedoch erscheint aufgrund der

höheren Betriebszuverlässigkeit und der Möglichkeiten effektiverer Programmorganisation auch der Einsatz von Halbleiterarbeitsspeichern als wünschenswert. Die Programmarchivierung und die Speicherung längerer Programme können dann mit einem sekundären Massenspeicher erfolgen, der aber weit weniger beansprucht wird als bei einem Einsatz als Operativspeicher der Steuerung.

Zur Senkung des Speicherplatzbedarfs bei gleicher Programmdauer bzw. zur Erhöhung der Programmdauer bei vorgegebenem Speicherumfang ist es sinnvoll, Verfahren zur Datenkompression vor der Abspeicherung in der Teach-in-Phase anzuwenden. Bei der Programmwiedergabe (Playback) ist dann das jeweils inverse Verfahren der Datendekompression anzuwenden. Dadurch entsteht das prinzipielle Verarbeitungsschema nach Bild 4.53. Dargestellt ist die Verarbeitung bei einer Achse. Nach der Abtastung und Digitalisierung der Eingangspositionswerte im Analog-Digital-Umsetzer ADU erfolgt eine Datenverdichtung mit dem Ziel einer effektiven Abspeicherung. Bei der Wiedergabe werden die Daten dekomprimiert und mit dem Digital-Analog-Umsetzer DAU in ein stufenförmiges Zeitsignal umgewandelt. Die endgültige Glättung erfolgt durch Trägheiten in den Servoregelkreisen. Das Verarbeitungsschema ist dem der digitalen Informationsübertragung sehr ähnlich, nur tritt dort an die Stelle des Arbeitsspeichers der Übertragungskanal. Deshalb sind die zum Einsatz kommenden Signalverarbeitungsverfahren mit denen der Übertragungstechnik verwandt bzw. direkt von dort übernommen (siehe unten).

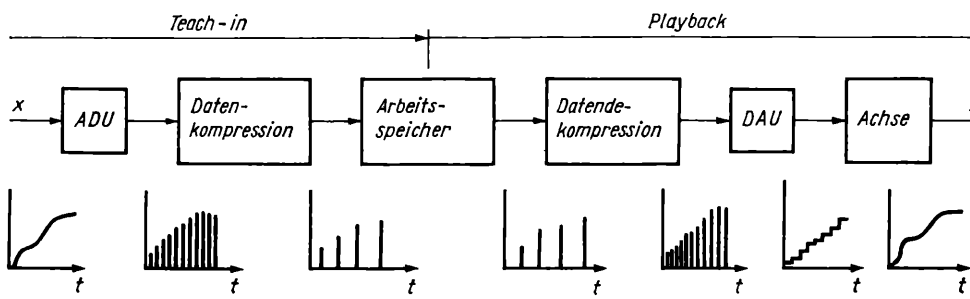


Bild 4.53. Signalverarbeitung in einer Multipunktsteuerung
ADU Analog-Digital-Umsetzer; *DAU* Digital-Analog-Umsetzer

Bei der Datenkompression sind Verfahren zur Reduzierung der *Speichertiefe* und der *Speicherwortbreite* zu unterscheiden.

In der ersten Gruppe wird das Ziel verfolgt, sowenig wie möglich Positionen abzuspeichern. Das ist

- durch eine reduzierte Abtastfrequenz beim Teach-in und einer Interpolation beim Playback oder
- durch eine adaptive Steuerung der Abtastfrequenz entsprechend der aktuellen Prozeßdynamik.

erreichbar.

Die zweite Gruppe von Verfahren geht davon aus, daß sich der Roboter stets mit begrenzter Geschwindigkeit und begrenzter Beschleunigung bewegt und deshalb Positionssprünge nicht auftreten können. Deshalb genügt es, verkürzte Informationen über aufeinanderfolgende Positionswerte aufzuzeichnen. Mögliche Verfahren sind hier das *Vorhersageverfahren* und die *Anwendung der Deltamodulation.*

Die genannten Verfahren sollen im folgenden kurz vorgestellt werden.

Zur Ermittlung der erforderlichen Abtastfrequenz bei der Aufnahme ist das Spektrum des abzutastenden Signals von Bedeutung. Bild 4.54 zeigt dazu Leistungsdichtespektren der Bewegung einer relativ langsamen Grundachse 1 und einer schnelleren Handachse 2 [4.28].

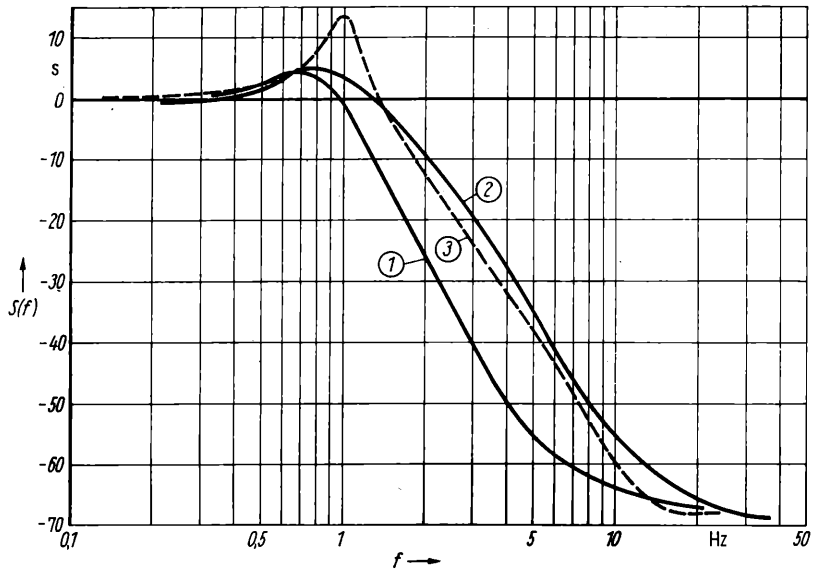


Bild 4.54. Leistungsspektrum der Achsbewegungen beim Farbspritzen
① für eine Grundachse [4.28]; ② für eine Handachse [4.28]; ③ Spektrum nach [4.27]

Zum Vergleich ist ein Leistungsdichtespektrum eingezeichnet, das in [4.27] zum Test von Datenkompressionsmethoden verwendet wurde. Die Spektren lassen sich erfahrungsgemäß recht gut durch Ausdrücke der Form

$$S(f) = \frac{1}{(1 + a_1 f^2 + a_2 f^4)(1 + a_3 f^2 + a_4 f^4)} + S_s \tag{4.112}$$

mit passend gewählten a_i; $i = 1$ bis 4 approximieren.

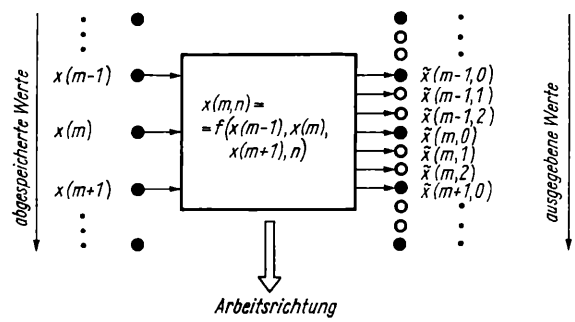


Bild 4.55
Arbeitsschema eines Interpolators

S_s ist ein Störspektrum, das in Näherung als konstant angenommen werden kann und etwa 70 dB unter $S(0)$ liegt. Man erkennt aus Bild 4.54, daß die Spektren bei Frequenzen von 10 bis 20 Hz in das Rauschen übergehen, woraus sich nach dem Abtasttheorem eine minimale Abtastfrequenz im Bereich von 20 bis 40 Hz für eine vollständige Rekonstruk-

tion des Ausgangssignals aus den Abtastwerten ergibt. Da in der Praxis immer gewisse Rekonstruktionsfehler zugelassen sind, kann mit noch geringeren Abtastfrequenzen gearbeitet werden (s. unten).

Da die Servoregelkreise Abtastfrequenzen von 50 bis 80 Hz für die Sollwertvorgabe erfordern, müssen Zwischenwerte zwischen den abgespeicherten Stützwerten bei der Wiedergabe ermittelt und eingefügt werden. Dies leistet der Interpolator, eine Funktionsgruppe, die heute vorwiegend programmtechnisch in einem Steuerrechner oder mit einem Spezialrechner innerhalb der Robotersteuerung realisiert wird. Ihre grundsätzliche Wirkungsweise ist im Bild 4.55 dargestellt. Der Interpolator liest eine bestimmte Anzahl (2 bis 5) aufeinanderfolgende Stützwerte aus dem Arbeitsspeicher aus, berechnet daraus die erforderliche Anzahl von Zwischenwerten und gibt diese an die Servoregler aus. Er arbeitet sich so Intervall für Intervall in Richtung zunehmender Zeit durch den Arbeitsspeicher hindurch, wobei zweckmäßig die einzelnen Achsen zyklisch durch das gleiche Interpolationsprogramm bedient werden.

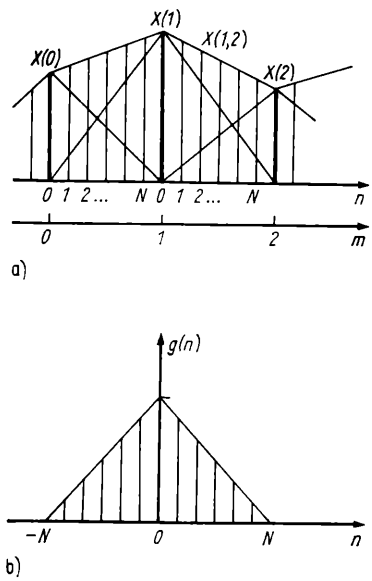
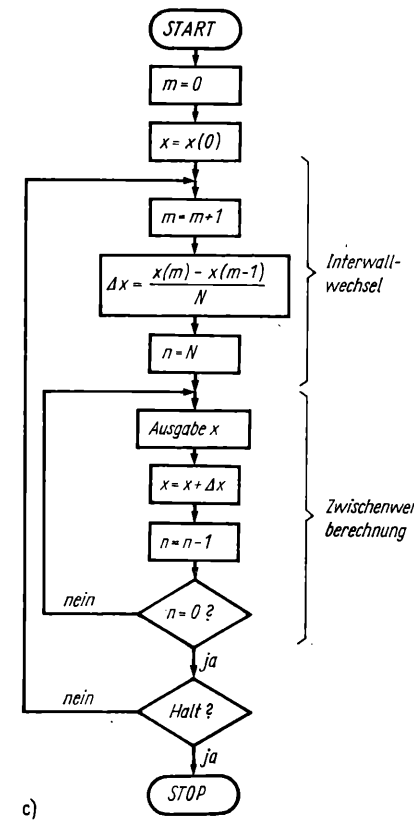


Bild 4.56. Linearinterpolation

a) Stützwerte und Zwischenwerte; b) Gewichtsfunktion eines Linearinterpolators; c) Algorithmus der Linearinterpolation



Am Beispiel der Linearinterpolation soll die Wirkungsweise genauer dargestellt werden. Im Bild 4.56a sind dazu die Stützwerte aus dem Arbeitsspeicher dick, die zu interpolierenden Zwischenwerte dünner gezeichnet; m ist dabei die laufende Nummer des Stützwertes, n die laufende Nummer des Zwischenwerts. Sollen N Zwischenwerte zwischen zwei Stützwerten eingefügt werden, so berechnen sie sich nach

$$x(m, n) = x(m)\left(1 - \frac{n}{N}\right) + x(m+1)\frac{n}{N}. \tag{4.113}$$

Ein Algorithmus, der diese Formel berechnet, ist im Bild 4.56c dargestellt. Er zeigt den typischen Aufbau vieler Interpolationsalgorithmen. In der inneren Schleife werden aufeinanderfolgend die Zwischenwerte eines Intervalls berechnet, während beim Übergang zum nächsten Intervall eine Vorbereitungsrechnung für die nächsten N Zwischenwerte (hier nur die Ermittlung von Δx) ausgeführt werden muß. Wählt man für N eine Potenz von 2, dann bedeutet die Division durch N bei digitaler Rechnung nur eine Rechtsverschiebung, und der dargestellte Algorithmus wird sehr schnell arbeiten.

Bei Betrachtung der Gl. (4.113) fällt auf, daß mit fortschreitendem n der Einfluß von $x(m)$ linear ab-, der von $x(m+1)$ linear zunimmt. Man kann sich die Entstehung des Geradenzugs im Bild 4.56a auch so vorstellen, daß die Folge der (dick hervorgehobenen) $x(m)$ mit der im Bild 4.56b gezeichneten Gewichtsfolge gefaltet wird. Diese Gewichtsfolge, die für die lineare Interpolation die Form

$$g(n) = \begin{cases} 1 - \dfrac{|n|}{N} & \text{für} \quad |n| < N \\ 0 & \text{sonst} \end{cases} \tag{4.114}$$

hat, kann als kennzeichnend für alle Interpolationsverfahren, die die Eigenschaft der Linearität besitzen, angesehen werden. Damit sind die Verfahren der Theorie zeitdiskreter Systeme zum Entwurf und zur Analyse von linearen Interpolationsalgorithmen anwendbar. Unter diesem Aspekt fällt auf, daß die Gewichtsfolge im Bild 4.56b für negative Werte von Null verschieden ist und somit ein akausales System beschreibt. Das stellt *hier* keine Einschränkung an die Realisierbarkeit dar, da der gesamte Prozeßverlauf gespeichert ist und somit auch auf die (von einem bestimmten m aus gesehene) Zukunft

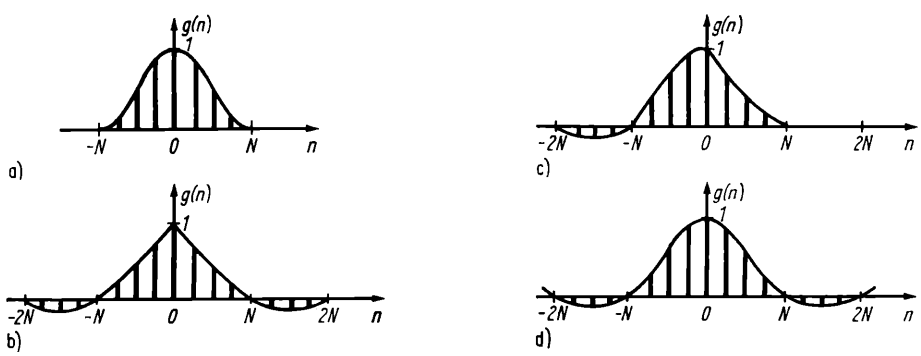


Bild 4.57. Gewichtsfunktionen von Interpolationsverfahren

a) Interpolation über ein Intervall mit stetigen Übergängen; b) quadratische Interpolation in disjunkten Intervallen; c) quadratische Interpolation mit überlappenden Intervallen; d) Interpolation durch ideale Bandbegrenzung

zugegriffen werden kann. Bild 4.57 zeigt die Gewichtsfunktionen einiger Interpolationsverfahren. Die Knickstellen des Polygonzugs im Bild 4.56a bedeuten in der Achsbewegung Sprünge der Geschwindigkeit. Soll das vermieden werden, kann ein Algorithmus mit einer Gewichtsfunktion nach Bild 4.57a verwendet werden. Er garantiert stetige Übergänge zwischen den Intervallen und stetigen Anschluß der Geschwindigkeiten. Rechentechnisch bietet die Interpolation durch Polynome besondere Vorteile. Dazu zeigen die Bilder 4.57b und 4.57c Gewichtsfunktionen von Algorithmen zur quadratischen Interpolation. Im ersten Fall (Bild 4.57b) wird durch je drei aufeinanderfolgende Stützwerte eine Parabel gelegt und verfahren, während im zweiten Fall (Bild 4.57c) die durch drei

aufeinanderfolgende Stützwerte gelegte Parabel jeweils nur im ersten Intervall verfahren wird. Damit wird die Größe des Geschwindigkeitssprungs von Bild 4.57b reduziert. Eine in gewisser Hinsicht optimale Interpolation wird bei Verwendung der Spaltfunktion

$$g(n) = \frac{\sin \pi \frac{n}{N}}{\pi \frac{n}{N}} \tag{4.115}$$

als Interpolatorgewichtsfolge erhalten. Sie entspricht einer idealen Frequenzbandbegrenzung des Eingangssignals. Bild 4.58 zeigt dazu den entstehenden Interpolationsfehler (mittlerer Fehlerbetrag) bei Anwendung dieser Interpolation auf die beim Farbspritzen beobachteten Prozesse.

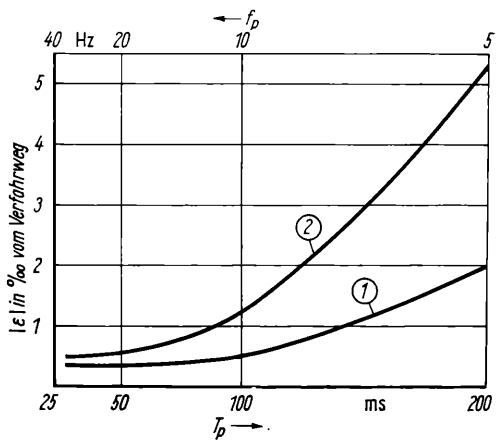


Bild 4.58. Mittlerer Fehlerbetrag bei Interpolation mit idealer Bandbegrenzung

(1) für die Grundachse nach Bild 4.54;
(2) für die Handachse nach Bild 4.54

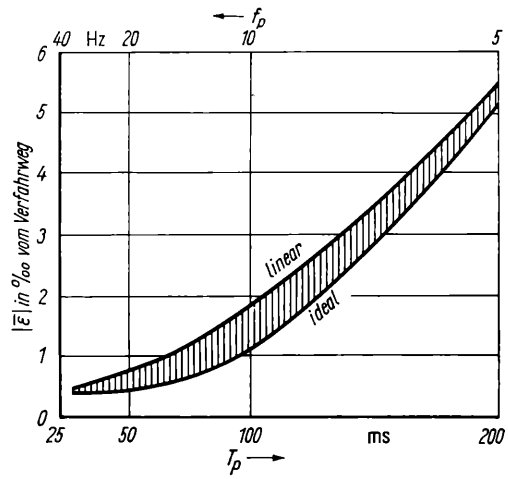


Bild 4.59. Mittlerer Fehlerbetrag bei verschiedenen Interpolationsverfahren in Abhängigkeit von der Abtastfrequenz beim Teach-in

Bild 4.59 gibt die Abhängigkeit des mittleren Fehlerbetrags von der Abtastfrequenz bei verschiedenen Interpolationsverfahren für den Bewegungsvorgang einer schnellen Handachse ((2) nach Bild 4.54 und Bild 4.58) an. Die Fehlerkurven für die im Bild 4.57 angegebenen Verfahren liegen im schraffierten Bereich.

Im Gegensatz zum bisher behandelten Verfahren, das die erforderliche Abtastfrequenz an den Prozeßeigenschaften als Ganzes bemißt, nimmt das Verfahren der adaptiven Steuerung der Abtastfrequenz auf die lokale Prozeßdynamik Bezug. Bild 4.60a zeigt das Wirkungsschema. In Abhängigkeit vom Prozeßverlauf wird dabei durch die Frequenzsteuerung ein Taktgenerator so gesteuert, daß bei hoher Prozeßdynamik (ein Maß dafür kann die aktuelle Geschwindigkeit sein [4.25]) eine hohe, bei langsamer Änderung der Positionswerte mit der Zeit eine niedrige Abtastfrequenz eingestellt wird. Zur einwandfreien Reproduktion der Positionswert-Zeit-Folge muß eine Information über die aktuelle Abtastfrequenz in den Arbeitsspeicher mit eingetragen werden. Diese steuert bei Wiedergabe den Taktgenerator, der physikalisch die gleiche Baugruppe wie bei der Aufnahme sein kann. Bild 4.60b zeigt ein Beispiel der adaptiven Taktung. Von den Verfahren zur Verminderung der Speicherwertbreite lassen sich viele in die Klasse der Vorhersage-

verfahren einordnen. Bild 4.61 a zeigt das Wirkungsschema. Durch ein Vorhersagefilter wird aus der Vergangenheit des Positions-Zeit-Verlaufs x ein Schätzwert x^* für den nächsten zu erwartenden Wert ermittelt und der Schätzfehler aufgezeichnet. Bei der Wiedergabe wird der eingegebene x-Wert-Verlauf durch ein entsprechendes inverses System (im Bild 4.61 a rechts dargestellt), das das gleiche Vorhersagefilter im Rückwärtszweig eines digitalen Integrators enthält, wieder rekonstruiert. Für das Vorhersagefilter können Algorithmen der polynomialen Vorhersage verwendet werden. Sie lauten für eine Vorhersage

$$\begin{aligned} &0.\ \text{Ordnung:}\ x^*(n) = x(n-1) \\ &1.\ \text{Ordnung:}\ x^*(n) = 2x(n-1) - x(n-2) \\ &2.\ \text{Ordnung:}\ x^*(n) = 3x(n-1) - 3x(n-2) + x(n-3). \end{aligned} \tag{4.116}$$

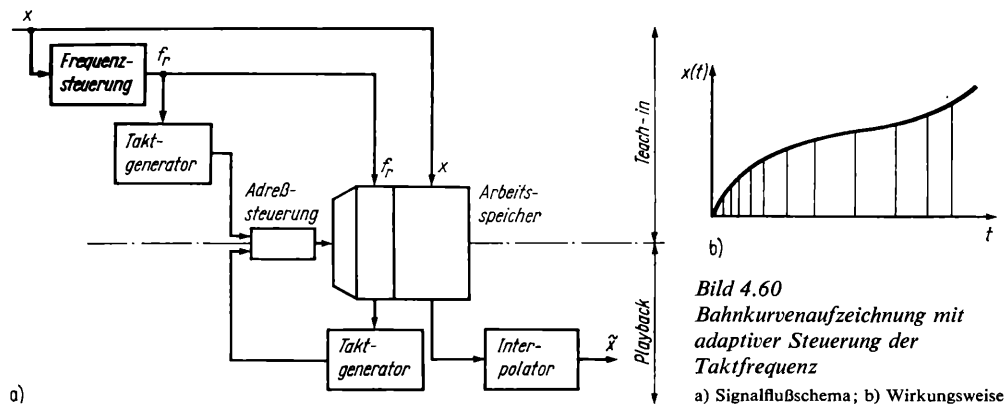


Bild 4.60
Bahnkurvenaufzeichnung mit adaptiver Steuerung der Taktfrequenz
a) Signalflußschema; b) Wirkungsweise

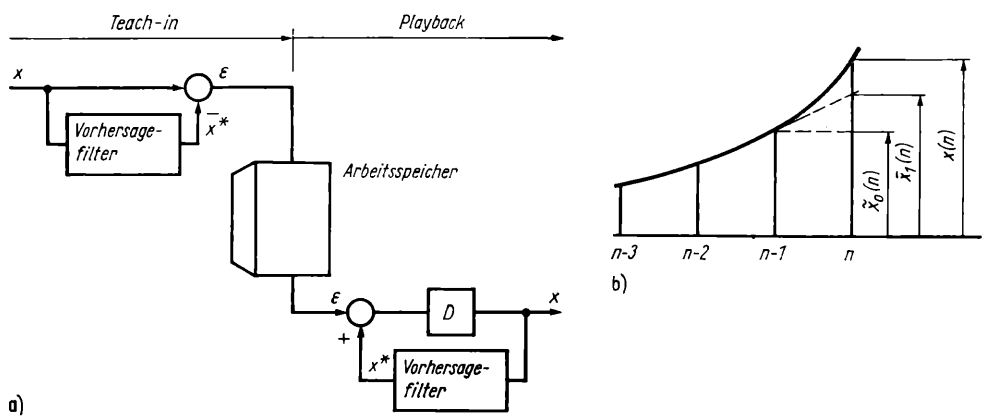


Bild 4.61. Bahnkurvenaufzeichnung mit Vorhersagefilter
a) Wirkungsschema; b) Vorhersage 0. und 1. Ordnung

Im ersten Fall wird angenommen, daß der Positionswert konstant bleibt und folglich die Differenz jeweils zweier aufeinanderfolgender Positionswerte aufgezeichnet wird. Bei der Vorhersage 1. Ordnung wird angenommen, daß der aus den beiden vergangenen x-Werten ermittelte lineare Trend anhält, und bei der Vorhersage 2. Ordnung wird der Schätzwert $x^*(n)$ als Wert einer Parabel, die durch die Werte $x(n-1)$, $x(n-2)$ und

$x(n-3)$ geht, berechnet. Das entspricht der Annahme einer gleichbleibenden Beschleunigung. Bild 4.61 b zeigt die Ermittlung des Schätzwerts $x^*(n)$ für eine Vorhersage 0. und 1. Ordnung. Anstelle der soeben dargestellten polynomialen Vorhersage kann auch eine optimale Vorhersage nach *Wiener* [4.29] vorgenommen werden, die auf der Auswertung des Leistungsdichtespektrums des Eingangsprozesses (Bild 4.54) basiert.

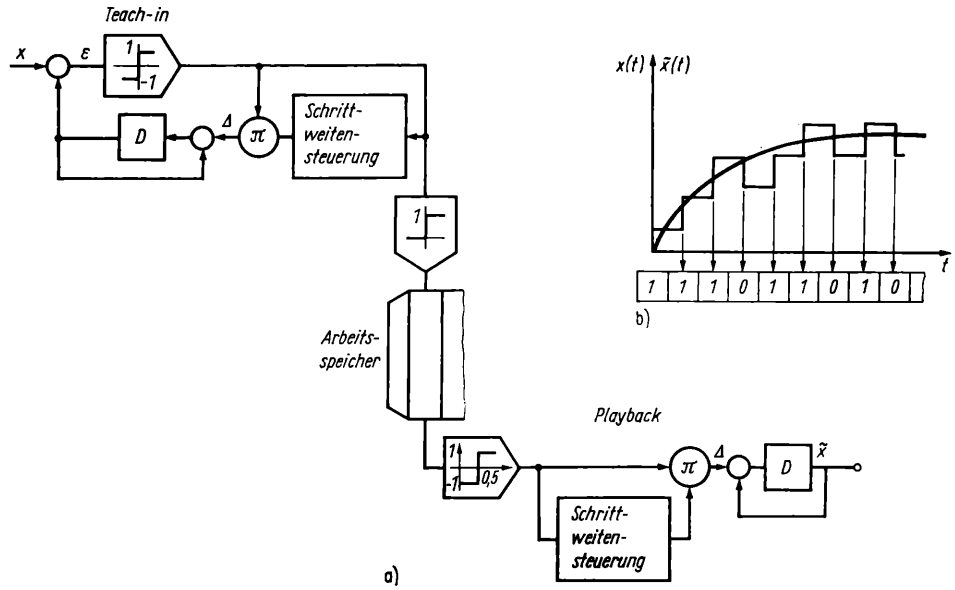


Bild 4.62. Bahnkurvenaufzeichnung mit Deltamodulation
a) Wirkungsschema; b) Zeitverläufe und abgespeicherte Bitfolge

Den Extremfall der Speicherwortverkürzung stellt das Datenkompressionsverfahren nach der Deltamodulation dar, das auch in der Übertragungstechnik häufig angewendet wird. Hier wird je Positionswert ein einziges Bit aufgezeichnet: allerdings muß die Abtastfrequenz viel höher liegen als bei den bisher behandelten Verfahren. Bild 4.62a zeigt das Wirkungsschema. Der aktuelle Positionswert x wird im Rhythmus des Aufnahmetakts mit einem Vergleichswert $\tilde{x}$ verglichen, der in einem digitalen Integrator nach der Vorschrift

$$\tilde{x} = \begin{cases} \tilde{x} + \Delta & \text{für} \quad \varepsilon \geqq 0 \\ \tilde{x} - \Delta & \text{für} \quad \varepsilon < 0 \end{cases} \tag{4.117}$$

gebildet wird, wobei

$$\varepsilon(n) = x(n) - \tilde{x}(n) \tag{4.118}$$

ist. Aufgezeichnet wird dabei

eine 1, wenn $\varepsilon \geqq 0$, und
eine 0, wenn $\varepsilon < 0$ ist.

Bild 4.62 b zeigt dazu einen Zeitverlauf $x(t)$, seine Approximation durch $\tilde{x}(t)$ und unten die aufgezeichnete Bitfolge. Für eine Achse wird damit eine Speicherbreite von 1 bit benötigt, und man kann acht Achsen in einem 1 byte breiten Speicher unterbringen. Dieses System ist aus der Regelungstechnik als Zweipunktregler bekannt. Die Reproduktion der

ursprünglichen Positionswert-Zeit-Folge geschieht wieder durch das inverse System, das in einem digitalen Integrator, von den abgespeicherten ε-Werten gesteuert, die entsprechenden Schrittwerte $+\Delta$ oder $-\Delta$ aufintegriert. Eine wesentliche Verbesserung des Verfahrens kann durch eine sich der Änderungstendenz des Verlaufs von x anpassende Schrittweitensteuerung von Δ erzielt werden. Sie ist im Bild 4.62a eingezeichnet.

4.6.2. Koordinatentransformation

Koordinatentransformationen für Industrieroboter stellen den funktionellen Zusammenhang zwischen natürlichen Koordinatenangaben und den Stellungen der Bewegungsachsen her. Position und Orientierung des Endeffektors sind i. allg. von der Stellung jedes einzelnen Manipulatorglieds abhängig. Bei n Gliedern muß jeder Raumpunkt durch n Positionswerte beschrieben werden; diese spannen ein n-dimensionales gerätespezifisches Koordinatensystem auf.

Bezüglich der Steuerung müssen Koordinatentransformationen zwei Aufgaben erfüllen:

- Bestimmung der Lage des Werkzeugs oder Greifers in kartesischen Koordinaten aus gegebenen Positionswerten der Dreh- und Schubgelenke,
- Berechnung der geräteeigenen Koordinaten aus den in kartesischen Koordinaten vorliegenden Steuerinformationen.

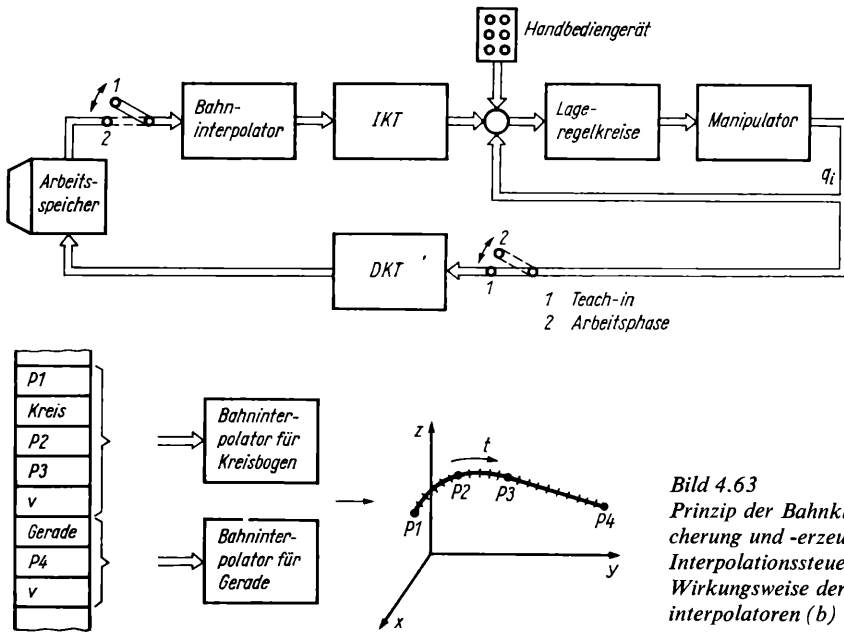


Bild 4.63
Prinzip der Bahnkurvenabspeicherung und -erzeugung bei der Interpolationssteuerung (a) und Wirkungsweise der Funktionsinterpolatoren (b)

Die erste Aufgabe ist von Bedeutung, wenn die zur Interpolation in kartesischen Koordinaten notwendigen Stützpunkte mittels Teach-in-Verfahrens gewonnen werden; sie soll als direkte Koordinatentransformation (DKT) bezeichnet werden. Die zweite Aufgabe ist nicht nur für die Umsetzung der Interpolationswerte bei Bahnsteuerungen notwendig; sie bildet auch eine Voraussetzung für die Eingabe von Steuerinformationen in natürlichen Koordinaten und die Nutzung von Sensorinformationen. Sie wird entsprechend der Aufgabenstellung des inversen kinematischen Problems [4.5] als inverse Ko-

ordinatentransformation (IKT) bezeichnet. Die Aufgaben der Koordinatentransformationen bei der Interpolationssteuerung verdeutlicht Bild 4.63. Die vom Effektor zu beschreibende Bahnkurve wird vom Bediener in Geradenstücke und Kreisbögen zerlegt, deren charakteristische Daten im indirekten Teach-in eingelesen werden.

Offensichtlich bestehen Echtzeitforderungen nur bezüglich der IKT. Da Koordinatentransformationen über diese beiden Aufgaben hinaus in vielfältiger Form zur Umrechnung zwischen verschiedenen ruhenden und bewegten Bezugssystemen und zur Aufgabenbeschreibung bei der textuellen Programmierung benötigt werden, sollen sie nachfolgend etwas ausführlicher dargestellt werden.

4.6.3. Homogene Transformationen

Einen allgemeinen Ansatz zum Aufstellen von Lösungsgleichungen für die Koordinatentransformation bietet die Transformation von Koordinatensystemen. Zwei orthonormierte Koordinatensysteme sind durch Translation und Rotation ineinander überführbar; es gelten folgende Zusammenhänge:

Die Koordinaten des Punktes $P(x', y', z')$ (Bild 4.64a) sind im Koordinatensystem, das aus dem gestrichenen durch Drehung um den Winkel Φ bei fester z-Achse entsteht,

$$\begin{aligned} x &= x' \cos \Phi + y' \sin \Phi \\ y &= -x' \sin \Phi + y' \cos \Phi \\ z &= z' \end{aligned} \tag{4.119}$$

oder

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos \Phi & \sin \Phi & 0 \\ -\sin \Phi & \cos \Phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \boldsymbol{R}_\Phi \boldsymbol{r}'. \tag{4.120}$$

Entsprechend gilt für die Drehung um die y'-Achse

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \cos \Theta & 0 & -\sin \Theta \\ 0 & 1 & 0 \\ \sin \Theta & 0 & \cos \Theta \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \boldsymbol{R}_\Theta \boldsymbol{r}' \tag{4.121}$$

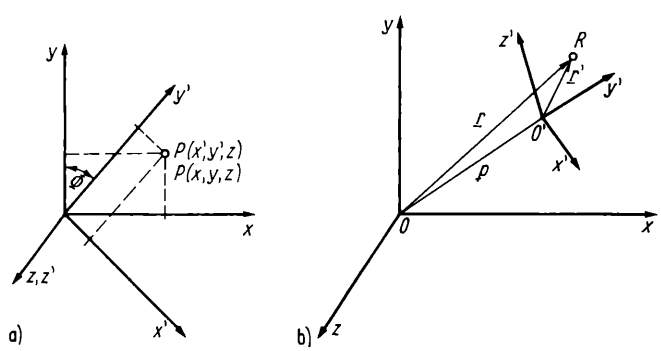


Bild 4.64. Drehung eines Koordinatensystems (a) und Darstellung eines ruhenden Koordinatensystems (b) in einem anderen, ebenfalls ruhenden System

und Rotation um die x'-Achse

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \boldsymbol{R}_\psi \boldsymbol{r}'. \tag{4.122}$$

Durch Multiplikation der Matrizen erhält man bei Beachtung der Reihenfolge die Gesamtrotationsmatrix

$$\boldsymbol{R}_{\Phi\Theta\psi} = \boldsymbol{R}_\Phi \boldsymbol{R}_\Theta \boldsymbol{R}_\psi = \begin{pmatrix} i'_x & j'_x & k'_x \\ i'_y & j'_y & k'_y \\ i'_z & j'_z & k'_z \end{pmatrix}. \tag{4.123}$$

Die Elemente der Rotationsmatrix sind die Richtungskosinus der gestrichenen Vektorbasis im ungestrichenen System:

$$\begin{aligned}
i'_x &= \cos\Phi\cos\Theta \\
j'_x &= \cos\Phi\sin\Theta\sin\psi - \sin\Phi\cos\psi \\
k'_x &= \sin\Phi\sin\psi - \cos\Phi\sin\Theta\cos\psi \\
i'_y &= -\cos\Theta\sin\Phi \\
j'_y &= \cos\Phi\cos\psi - \sin\Phi\sin\Theta\sin\psi \\
k'_y &= \cos\Phi\sin\psi + \sin\Phi\sin\Theta\cos\psi \\
i'_z &= \sin\Theta \\
j'_z &= -\sin\psi\cos\Theta \\
k'_z &= \cos\Theta\cos\psi .
\end{aligned} \tag{4.124}$$

Anschaulich bedeuten sie die Projektionen der Einheitsvektoren $\boldsymbol{i}', \boldsymbol{j}', \boldsymbol{k}'$ auf die entsprechenden Koordinatenachsen des ungestrichenen Systems. Entsprechend Bild 4.64b läßt sich damit der Punkt $R(x', y', z')$, dessen Koordinaten im gestrichenen System vorliegen, im ungestrichenen angeben zu

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} + \boldsymbol{R}_{\Phi\Theta\psi} \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \boldsymbol{p} + \boldsymbol{R}_{\Phi\Theta\psi}\boldsymbol{r}'. \tag{4.125}$$

Gl. 4.125 läßt sich in folgender Form zusammenfassen:

$$\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} i'_x & j'_x & k'_x & p_x \\ i'_y & j'_y & k'_y & p_y \\ i'_z & j'_z & k'_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \boldsymbol{T}\boldsymbol{r}'. \tag{4.126}$$

Diese Darstellung hat in der projektiven Geometrie Bedeutung. Durch die Beschreibung eines Punktes mit vier Koordinaten

$$P\left(\frac{x}{k}, \frac{y}{k}, \frac{z}{k}\right) = P(x, y, z, k)$$

wird in der obigen Transformation eine Gesamtskalierung möglich; zur maßstabgerechten Umrechnung ist $k = 1$ zu setzen. Die Werte x, y, z, k werden als homogene Koordinaten bezeichnet, T als homogene Transformation. Häufig werden anstelle der Drehwinkel um die Achsen des Koordinatensystems auch die Eulerschen Winkel benutzt; die Richtungskosinus werden damit

$$
\begin{aligned}
i'_x &= \cos\Phi_e \cos\Theta_e \cos\psi_e - \sin\Phi_e \sin\psi_e \\
j'_x &= -\cos\Phi_e \cos\Theta_e \sin\psi_e - \sin\Phi_e \cos\psi_e \\
k'_x &= \cos\Phi_e \sin\Theta_e \\
i'_y &= \sin\Phi_e \cos\Theta_e \cos\psi_e + \cos\Phi_e \sin\psi_e \\
j'_y &= -\sin\Phi_e \cos\Theta_e \sin\psi_e + \cos\Phi_e \cos\psi_e \\
k'_y &= \sin\Phi_e \sin\Theta_e \\
i'_z &= -\sin\Theta_e \cos\psi_e \\
j'_z &= \sin\Theta_e \sin\psi_e \\
k'_z &= \cos\Theta_e .
\end{aligned}
\tag{4.127}
$$

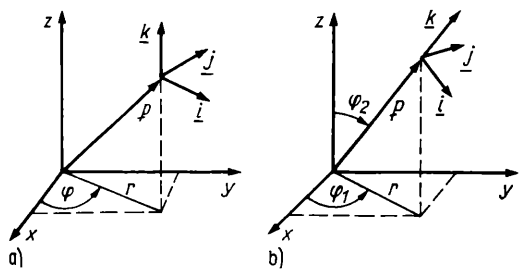


Bild 4.65
Darstellung eines Koordinatensystems, das in zylindrischen (a) und in sphärischen Koordinaten (b) bezüglich des Basissystems verschoben ist

Entsteht die Vektorbasis des gestrichenen Systems durch Verschiebung und Drehung in zylindrischen Koordinaten aus dem ursprünglichen, gilt die Transformation (Bild 4.65)

$$
\boldsymbol{T}_z = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 & r\cos\varphi \\ \sin\varphi & \cos\varphi & 0 & r\sin\varphi \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix}
\tag{4.128}
$$

und für Verschiebung in Kugelkoordinaten r, φ_1, φ_2 ($r = |\boldsymbol{p}|$)

$$
\boldsymbol{T}_k = \begin{pmatrix} \cos\varphi_1 \cos\varphi_2 & -\sin\varphi_1 & \cos\varphi_1 \sin\varphi_2 & r\cos\varphi_1 \sin\varphi_2 \\ \sin\varphi_1 \cos\varphi_2 & \cos\varphi_1 & \sin\varphi_1 \sin\varphi_2 & r\sin\varphi_1 \sin\varphi_2 \\ -\sin\varphi_2 & 0 & \cos\varphi_2 & r\cos\varphi_2 \\ 0 & 0 & 0 & 1 \end{pmatrix} .
\tag{4.129}
$$

4.6.4. Direkte Koordinatentransformation

Die Angabe von Position und Orientierung des Endeffektors hängt wesentlich vom vorgesehenen Einsatzfall ab. Eine zweckmäßige Wahl des Positionspunktes P (x, y, z) für einen Greifer wird im Bild 4.66a gezeigt. Legt man in diesen Bezugspunkt ein greifer-

festes Koordinatensystem, läßt sich die Orientierung durch geeignete Wahl eines zweiten Punktes in diesem Koordinatensystem angeben. Beide Punkte beschreiben eine Gerade im Raum, deren Richtung im raumfesten Basissystem x, y, z die Orientierung festlegt. Ein solcher Punkt im Greiferkoordinatensystem ist auch der Punkt R' (1, 1, 1), der durch die Einheitsvektoren i_g, j_g, k_g definiert ist [4.5]. Position und Orientierung sind damit durch die homogene Transformationsmatrix

$$\boldsymbol{T}_g = \begin{pmatrix} i_{gx} & j_{gx} & k_{gx} & p_x \\ i_{gy} & j_{gy} & k_{gy} & p_y \\ i_{gz} & j_{gz} & k_{gz} & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.130}$$

angebbar. Wegen $\boldsymbol{i}_g = \boldsymbol{j}_{gx}\boldsymbol{k}_g$ sind die ersten drei Spalten nicht voneinander unabhängig.

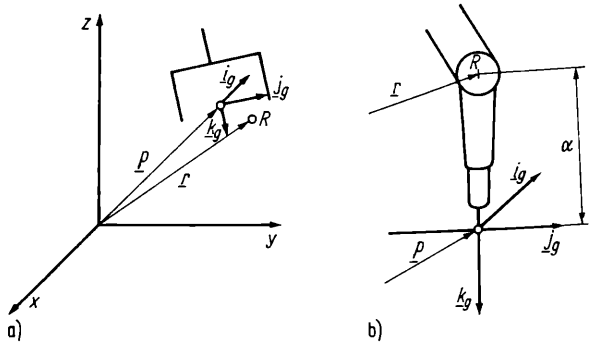


Bild 4.66. Effektorkoordinatensystem für Greifer (a) und Schweißkopf (b)

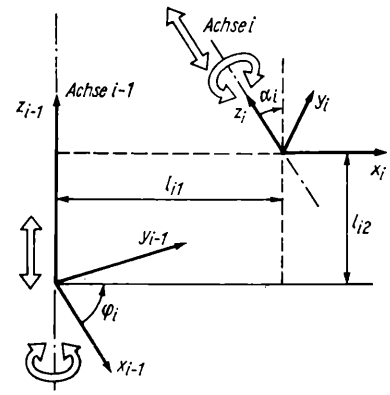


Bild 4.67. Zur Ableitung der Denavit-Hartenberg-Matrix

Die Orientierung ist durch Angabe von drei Elementen der Rotationsmatrix eindeutig bestimmt, wird jedoch eines dieser Elemente bei bestimmten Greiferstellungen Null, treten Mehrdeutigkeiten auf. Mit den Beziehungen (4.124) lassen sich die Orientierungswinkel aus $\boldsymbol{T}_g$ ableiten:

$$\Phi = \arctan -\frac{i_{gy}}{i_{gx}}, \qquad \psi = \arctan -\frac{j_{gz}}{k_{gz}}, \qquad \Theta = \arcsin i_{gz}. \tag{4.131}$$

Für einen rotationssymmetrischen Schweißkopf gemäß Bild 4.66b läßt sich die Orientierung aus der Transformation

$$\boldsymbol{T}_s = \boldsymbol{T}_g \begin{pmatrix} 0 \\ 0 \\ -a \\ 1 \end{pmatrix} \tag{4.132}$$

bestimmen. Der Vergleich mit Gl. (4.124) zeigt, daß hierbei der Winkel Φ unbestimmt bleibt, da R auf der z'-Achse liegt. Wir setzen $\Phi = 0$ und erhalten als Orientierungswinkel

$$\psi = \arcsin k_{gy}, \qquad \Theta = \arctan -\frac{k_{gx}}{k_{gz}}. \tag{4.133}$$

Für die Effektorstellung ergeben sich damit in Komponenten des Basissystems x, y, z

$$\begin{aligned} r_x &= i_{gx} + j_{gx} + k_{gx} + p_x \\ r_y &= i_{gy} + j_{gy} + k_{gy} + p_y \\ r_z &= i_{gz} + j_{gz} + k_{gz} + p_z . \end{aligned} \tag{4.134}$$

Diese Darstellung läßt sich vorteilhaft bei der indirekten Teach-in-Programmierung zum Verfahren der Roboterachsen in natürlichen Koordinaten anwenden. Ordnet man das Handbediengerät im Bild 4.63 vor der IKT an, sind folgende Handbedienfunktionen realisierbar:

- Verfahren in den Richtungen x, y, z bei raumfester Orientierung; Eingabedaten p_x, p_y, p_z,
- Änderung der Orientierung im greifereigenen Koordinatensystem bei fester Position durch Rotation um die Achsen x_g, y_g, z_g; die Eingabedaten berechnen sich aus

$$\begin{pmatrix} i_{gx} & j_{gx} & k_{gx} \\ i_{gy} & j_{gy} & k_{gy} \\ i_{gz} & j_{gz} & k_{gz} \end{pmatrix} \underset{\substack{\vdots \\ (\text{bzw. } \boldsymbol{R}_\Theta,\, \boldsymbol{R}_\psi)}}{\boldsymbol{R}_\Phi} - \begin{pmatrix} i_{gx} & j_{gx} & k_{gx} \\ i_{gy} & j_{gy} & k_{gy} \\ i_{gz} & j_{gz} & k_{gz} \end{pmatrix} = \Delta R; \qquad \begin{aligned} \Delta r_x &= \Delta i_{gx} + \Delta j_{gx} + \Delta k_{gx} \\ \Delta r_y &= \Delta i_{gy} + \Delta j_{gy} + \Delta k_{gy} \\ \Delta r_z &= \Delta i_{gz} + \Delta j_{gz} + \Delta k_{gz} , \end{aligned} \tag{4.135}$$

- Änderung der Orientierung bei fester Position durch Rotation um die Achsen des Basissystems; die Eingabedaten berechnen sich aus

$$\underset{\substack{\vdots \\ (\text{bzw. } \boldsymbol{R}_\Theta,\, \boldsymbol{R}_\psi)}}{\boldsymbol{R}_\Phi} - \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \Delta r, \qquad \begin{aligned} \Delta r_x &= \Delta i_{gx} \\ \Delta r_y &= \Delta j_{gy} \\ \Delta r_z &= \Delta k_{gz} . \end{aligned} \tag{4.136}$$

Um die Elemente der Matrix $\boldsymbol{T}_g$ aus der momentanen Stellung der n Gelenke zu bestimmen, ordnen wir jedem Gelenk ein gliedfestes Koordinatensystem zu, so daß das n-te Koordinatensystem mit dem werkzeugeigenen Koordinatensystem übereinstimmt (Bild 4.66). Unter der Voraussetzung, daß die Achse z_{i-1} in der Bewegungsachse des folgenden Gliedes liegt und $x_i \perp z_{i-1}$ und von der Achse $i - 1$ fortzeigend, läßt sich das i-te Koordinatensystem im Koordinatensystem $i - 1$ durch die Denavit-Hartenberg-Matrix [4.30] beschreiben (Bild 4.67):

$$\boldsymbol{A}_i = \begin{pmatrix} \cos\varphi_i & -\cos\alpha_i \sin\varphi_i & \sin\alpha_i \sin\varphi_i & l_{i1} \cos\varphi_i \\ \sin\varphi_i & \cos\alpha_i \cos\varphi_i & -\sin\alpha_i \cos\varphi_i & l_{i1} \sin\varphi_i \\ 0 & \sin\alpha_i & \cos\alpha_i & l_{i2} \\ 0 & 0 & 0 & 1 \end{pmatrix} . \tag{4.137}$$

Sie stellt eine reduzierte Form der homogenen Transformation dar, da unter den obigen Bedingungen die Koordinatensysteme durch zwei Translationen l_{i1}, l_{i2} und zwei Rotationen φ, α ineinander übergeführt werden. Industrierobotergelenke werden in der Regel so konstruiert, daß die $\alpha_i = 0°$ bzw. $90°$ sind und eine der Translationen Null ist, was zu weiteren Vereinfachungen führt. Bei Schubgelenken ist eines der l_i variabel, bei Drehgelenken φ_i. Liegt im weiteren nur eine Translation vor, wird sie mit l_i bezeichnet.

Die Greiferstellung T_g erhält man nun in Basiskoordinaten durch Multiplikation aller A-Matrizen

$$\boldsymbol{T}_g = \boldsymbol{A}_1 \boldsymbol{A}_2 \ldots \boldsymbol{A}_n . \tag{4.138}$$

Für die Konfiguration im Bild 4.68 ergeben sich die A-Matrizen zu

$$
\begin{aligned}
A_1 &= \begin{pmatrix} \cos\varphi_1 & 0 & \sin\varphi_1 & 0 \\ \sin\varphi_1 & 0 & -\cos\varphi_1 & 0 \\ 0 & 1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
A_2 &= \begin{pmatrix} \cos\varphi_2 & -\sin\varphi_2 & 0 & l_2\cos\varphi_2 \\ \sin\varphi_2 & \cos\varphi_2 & 0 & l_2\sin\varphi_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
A_3 &= \begin{pmatrix} \cos\varphi_3 & -\sin\varphi_3 & 0 & l_3\cos\varphi_3 \\ \sin\varphi_3 & \cos\varphi_3 & 0 & l_3\sin\varphi_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
A_4 &= \begin{pmatrix} \cos\varphi_4 & 0 & \sin\varphi_4 & 0 \\ \sin\varphi_4 & 0 & -\cos\varphi_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
A_5 &= \begin{pmatrix} \cos\varphi_5 & 0 & -\sin\varphi_5 & 0 \\ \sin\varphi_5 & 0 & \cos\varphi_5 & 0 \\ 0 & -1 & 0 & l_5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
A_6 &= \begin{pmatrix} \cos\varphi_6 & -\sin\varphi_6 & 0 & 0 \\ \sin\varphi_6 & \cos\varphi_6 & 0 & 0 \\ 0 & 0 & 1 & l_6 \\ 0 & 0 & 0 & 1 \end{pmatrix}.
\end{aligned}
\tag{4.139}
$$

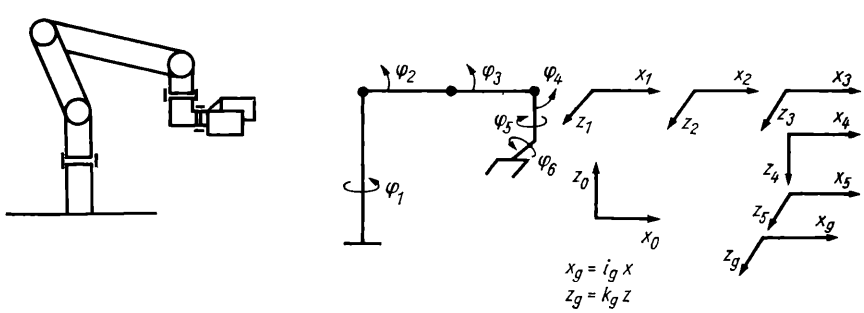


Bild 4.68. 6-Achsen-Gelenkroboter und dessen kinematisches Schema mit gelenkeigenen Koordinatensystemen

Durch Ausmultiplizieren von rechts nach links erhalten wir die Bestimmungsgleichungen für die Elemente der Matrix $\boldsymbol{T}_g$:

$$
\begin{aligned}
i_{gx} &= \cos\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \cos\varphi_6 - \sin\varphi_1 \sin\varphi_5 \cos\varphi_6 \\
&\quad - \sin(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_1 \sin\varphi_6 \\
i_{gy} &= \sin\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \cos\varphi_6 + \cos\varphi_1 \sin\varphi_5 \cos\varphi_6 \\
&\quad - \sin(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_1 \sin\varphi_6 \\
i_{gz} &= \sin(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \cos\varphi_6 + \cos(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_6 \\
j_{gx} &= -\cos\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \sin\varphi_6 - \sin\varphi_1 \sin\varphi_5 \sin\varphi_6 \\
&\quad - \cos\varphi_1 \sin(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_6 \\
j_{gy} &= -\sin\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \sin\varphi_6 + \cos\varphi_1 \sin\varphi_5 \sin\varphi_6 \\
&\quad - \sin\varphi_1 \sin(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_6 \\
j_{gz} &= \cos(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_6 - \sin(\varphi_2 + \varphi_3 + \varphi_4) \cos\varphi_5 \sin\varphi_6 \\
k_{gx} &= \sin\varphi_1 \cos\varphi_5 - \cos\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5 \qquad (4.140) \\
k_{gy} &= -\sin\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5 - \cos\varphi_1 \cos\varphi_5 \\
k_{gz} &= -\sin(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5 \\
p_x &= l_2 \cos\varphi_1 \cos\varphi_2 + l_3 \cos\varphi_1 \cos(\varphi_2 + \varphi_3) + l_5 \sin(\varphi_2 + \varphi_3 + \varphi_4) \\
&\quad \cos\varphi_1 + l_6 [\sin\varphi_1 \cos\varphi_5 - \cos\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5] \\
p_y &= l_2 \sin\varphi_1 \cos\varphi_2 + l_3 \sin\varphi_1 \cos(\varphi_2 + \varphi_3) + l_5 \sin\varphi_1 \sin(\varphi_2 + \varphi_3 + \varphi_4) \\
&\quad - l_6 [\sin\varphi_1 \cos(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5 + \cos\varphi_1 \cos\varphi_5] \\
p_z &= l_1 + l_2 \sin\varphi_2 + l_3 \sin(\varphi_2 + \varphi_3) - l_5 \cos(\varphi_2 + \varphi_3 + \varphi_4) \\
&\quad - l_6 [\sin(\varphi_2 + \varphi_3 + \varphi_4) \sin\varphi_5].
\end{aligned}
$$

Bei der Festlegung der Nullstellung für die Gelenkwinkel ist zu beachten, daß keine Mehrdeutigkeiten auftreten. Die i_g-Komponenten lassen sich auch aus dem Vektorprodukt der zweiten und dritten Spalte der Matrix $\boldsymbol{T}_g$ ermitteln, so daß die Bestimmung von sechs Rotationselementen ausreicht. Die Orientierungswinkel lassen sich mit Hilfe der Beziehungen (4.131) berechnen.

Verhältnismäßig einfache Beziehungen für die DKT ergeben sich, wenn sich die Orientierung des Endeffektors relativ zur Position der Grundachsen durchführen läßt oder einschränkende Bedingungen für die Orientierung vorliegen, z. B. senkrechte Greiferstellung. Unter bestimmten Voraussetzungen für die Anordnung der Handachsen läßt sich die Koordinatentransformation generell auf eine getrennte Betrachtung der Hand- und der Grundachsen zurückführen, was beträchtliche Vereinfachungen mit sich bringt [4.31].

Für zylindrische Grundachsen DLL (Bild 3.2) berechnet sich die Position des Endpunktes der dritten Achse aus

$$\begin{aligned} p_x &= l_2 \cos \varphi_1 \\ p_y &= l_2 \sin \varphi_1 \\ p_z &= l_1 \end{aligned} \tag{4.141}$$

für Kugelkoordinaten aus Gl. (4.129)

$$\begin{aligned} p_x &= l_1 \cos \varphi_1 \sin \varphi_2 \\ p_y &= l_1 \sin \varphi_1 \sin \varphi_2 \\ p_z &= l_1 \cos \varphi_2 \end{aligned} \tag{4.142}$$

und für die Struktur DDD (Bild 4.69c)

$$\begin{aligned} p_x &= l_2 \cos \varphi_1 \cos \varphi_2 + l_3 \cos (\varphi_2 + \varphi_3) \cos \varphi_1 \\ p_y &= l_2 \cos \varphi_2 \sin \varphi_1 + l_3 \cos (\varphi_2 + \varphi_3) \sin \varphi_1 \\ p_z &= l_1 + l_2 \sin \varphi_2 - l_3 \sin (\varphi_2 + \varphi_3). \end{aligned} \tag{4.143}$$

Die Realisierung der DKT ist nicht an Echtzeitforderungen gebunden, da sie nur in der Teach-in-Phase benötigt wird. Die trigonometrischen Funktionen werden entweder durch Tabellenzugriff oder mittels Standardroutinen bestimmt.

4.6.5. Inverse Koordinatentransformation

Die IKT berechnet aus gegebenen kartesischen Koordinaten p_x, p_y, p_z und r_x, r_y, r_z die zugehörigen Gelenkstellungen

$$\begin{aligned} q_1 &= f_1(\boldsymbol{p}, \boldsymbol{r}) \\ &\vdots \\ q_n &= f_n(\boldsymbol{p}, \boldsymbol{r}). \end{aligned} \tag{4.144}$$

Industrieroboter sind in der Regel so konstruiert, daß dieses Gleichungssystem für $n \leqq 6$ im klassischen Sinne lösbar ist. Ist dies nicht der Fall, nennt man die Konstruktion redundant. Zur Lösung der inversen Koordinatentransformation für redundante Konfigurationen sind zusätzliche Bestimmungsgleichungen erforderlich, die aus der durchzuführenden Bewegung gewonnen werden. Für den redundanten Fall, daß drei Drehgelenke φ_1, φ_1, φ_3 in einer Ebene liegen, läßt sich durch die Annahme, daß $\varphi_1 = \frac{1}{2} \times (\varphi_1 + \varphi_2)$, eine einfache Zusatzbedingung schaffen, die die Lösbarkeit garantiert.

Die Lösungsgleichungen lassen sich sukzessive aus Gl. (4.138) durch Multiplikation mit inversen A-Matrizen in folgender Reihenfolge aufstellen:

$$\begin{aligned} \boldsymbol{A}_2 \boldsymbol{A}_3 \ldots \boldsymbol{A}_n &= \boldsymbol{A}_1^{-1} \boldsymbol{T}_g \\ \boldsymbol{A}_3 \boldsymbol{A}_4 \ldots \boldsymbol{A}_n &= \boldsymbol{A}_2^{-1} \boldsymbol{A}_1^{-1} \boldsymbol{T}_g \\ &\vdots \\ \boldsymbol{A}_n &= \boldsymbol{A}_{n-1}^{-1} \ldots \boldsymbol{A}_1^{-1} \boldsymbol{T}_g. \end{aligned} \tag{4.145}$$

Da jede der A-Matrizen nur eine Variable enthält, ergeben sich aus den rechten Seiten nacheinander n Lösungsgleichungen durch Elementevergleich der rechten und linken

Seiten und Eliminieren der noch unbekannten Gelenkkoordinaten. Die Determinanten der A-Matrizen sind dabei stets eins. Für die Konfiguration Bild 4.69 ergeben sich mit den A-Matrizen aus Gl. (4.139) folgende Lösungen für die Gelenkkoordinaten:

$$\varphi_1 = \arctan\left(\frac{p'_y}{p'_x}\right), \qquad \boldsymbol{p}' = (p'_x, p'_y, p'_z) = \boldsymbol{p} - l_6 \boldsymbol{k}_g$$

$$(\varphi_2 + \varphi_3 + \varphi_4) = \arctan \frac{k_{gz}}{\cos\varphi_1 k_{gx} + \sin\varphi_1 k_{gy}} \tag{4.146}$$

$$\varphi_3 = \arctan \frac{\sin\varphi_3}{\cos\varphi_3}$$

$$\cos\varphi_3 = \frac{\begin{array}{l}[\cos\varphi_1 p'_x + \sin\varphi_1 p'_y - \sin(\varphi_2 + \varphi_3 + \varphi_4) l_5]^2 \\ + [p'_z + \cos(\varphi_2 + \varphi_3 + \varphi_4) l_5]^2 - l_2^2 - l_3^2\end{array}}{2 l_2 l_3}$$

$$\sin\varphi_3 = \pm\sqrt{1 - \cos^2\varphi_3}$$

$$\varphi_2 = \arctan \frac{\begin{array}{l}(l_3\cos\varphi_3 + l_2)[p'_z + \cos(\varphi_2 + \varphi_3 + \varphi_4) l_5] \\ - l_3 \sin\varphi_3 [\cos\varphi_1 p'_x + \sin\varphi_1 p'_y - l_5 \sin(\varphi_2 + \varphi_3 + \varphi_4)]\end{array}}{\begin{array}{l}(l_3\cos\varphi_3 + l_2)[\cos\varphi_1 p'_x + \sin\varphi_1 p'_y - \sin(\varphi_2 + \varphi_3 + \varphi_4) l_5] \\ + l_3 \sin\varphi_3 [p'_z + \cos(\varphi_2 + \varphi_3 + \varphi_4) l_5]\end{array}}$$

$$\varphi_4 = (\varphi_2 + \varphi_3 + \varphi_4) - \varphi_3 - \varphi_2$$

$$\varphi_5 = \arctan \frac{\begin{array}{l}-\cos(\varphi_2 + \varphi_3 + \varphi_4)(\cos\varphi_1 k_{gx} + \sin\varphi_1 k_{gy}) \\ - \sin(\varphi_2 + \varphi_3 + \varphi_4) k_{gz}\end{array}}{\sin\varphi_1 k_{gx} - \cos\varphi_1 k_{gx}}$$

$$\varphi_6 = \arctan \frac{\begin{array}{l}-\cos\varphi_5 [\cos(\varphi_2 + \varphi_3 + \varphi_4)(\cos\varphi_1 j_{gx} + \sin\varphi_1 j_{gy}) \\ + \sin(\varphi_2 + \varphi_3 + \varphi_4) j_{gz}] - \sin\varphi_5 (\sin\varphi_1 j_{gx} - \cos\varphi_1 j_{gy})\end{array}}{\begin{array}{l}\cos(\varphi_2 + \varphi_3 + \varphi_4) j_{gz} - \sin(\varphi_2 + \varphi_3 + \varphi_4)(j_{gx}\cos\varphi_1 \\ + j_{gy}\sin\varphi_1)\end{array}} .$$

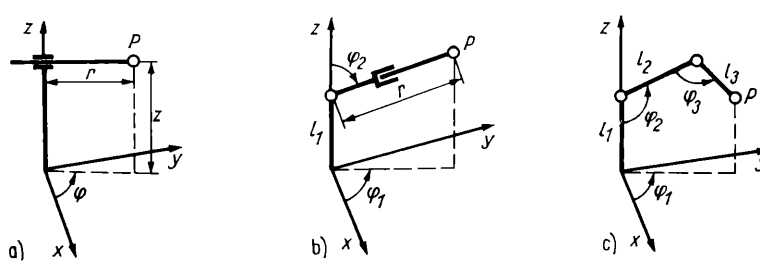


Bild 4.69. Zylindrische (a), sphärische (b) und Dreigelenk-Grundachsen (c)
Der Positionspunkt ist hierbei auf das Ende der letzten Grundachse bezogen.

Liegen weniger als sechs Gelenke vor, ist das Koordinatensystem des jeweils letzten Gelenkes in den Effektor zu legen; die Achsbezeichnungen dieses Koordinatensystems sind entsprechend den Forderungen an die A-Matrizen zu wählen. Entfällt beispielsweise das sechste Drehgelenk, d.h. $\varphi_6 = 0$, gilt für das neue Effektorsystem der fünften Achse $\boldsymbol{k}_g = -\boldsymbol{j}_{g5}, \boldsymbol{j}_g = \boldsymbol{i}_{g5}, \boldsymbol{i}_g = -\boldsymbol{k}_{g5}$, wobei $\boldsymbol{i}_{g5}, \boldsymbol{j}_{g5}, \boldsymbol{k}_{g5}$ die Vektorbasis des Koordinaten-

systems x_5, y_5, z_5 sind. Zur effektiven Berechnung sind aus Gl. (4.146) mehrfach benutzte Ausdrücke als Blöcke zusammenzufassen.

Aus den möglichen Bestimmungsgleichungen ist die arctan-Darstellung vorzuziehen, da sie innerhalb der in Frage kommenden Wertebereiche auf einem Rechner eindeutig bestimmbar ist.

Für die Grundachsen DDD gemäß Bild 4.69c erhält man aus Gl. (4.143) die Gelenkkoordinaten

$$\begin{aligned}
\varphi_1 &= \arctan \frac{p_y}{p_x} \\
\varphi_2 &= \arctan \left(\frac{p_z}{\sqrt{p_x^2 + p_y^2}} \right) + \arccos \left(\frac{l_2^2 - l_3^2 + p_x^2 + p_y^2 + p_z^2}{2 l_2 \sqrt{p_x^2 + p_y^2 + p_z^2}} \right) \\
\varphi_3 &= \arccos \left(\frac{l_2^2 + l_3^2 - p_x^2 - p_y^2 - p_z^2}{2 l_2 l_3} \right).
\end{aligned} \tag{4.147}$$

Die bisher dargestellten Verfahren berechnen die absoluten Werte der Gelenkkoordinaten aus den kartesischen Koordinaten. Für Bahnsteuerungen muß die IKT eine Punktfolge $\boldsymbol{s}(\Delta t_i)$, $\boldsymbol{s} = (r_x, r_y, r_z, p_x, p_y, p_z)^{\mathrm{T}}$ in die Sollwertfolge $\boldsymbol{q}(\Delta t_i)$, $\boldsymbol{q} = (q_1, \ldots, q_n)^{\mathrm{T}}$ umrechnen. Sind die Δt_i ausreichend klein, gilt für zwei aufeinanderfolgende Bahnpunkte $\boldsymbol{s}_i \approx \boldsymbol{s}_{i+1}$, da aufgrund der begrenzten Beschleunigung keine Sprünge in der Bewegungsbahn auftreten können. Wird nur die Abweichung $\Delta \boldsymbol{s}_i = \boldsymbol{s}_i - \boldsymbol{s}_{i+1}$ in die entsprechende Abweichung der Sollwertfolge $\boldsymbol{q}_i$ umgerechnet, entsteht ein *inkrementelles Verfahren*. Für den Bahnvektor in kartesischen Koordinaten gilt

$$\boldsymbol{s}(t) = \boldsymbol{s}\,[q_1(t), \ldots, q_n(t)].$$

Unter der Voraussetzung, daß $\boldsymbol{s}(t)$ und $\boldsymbol{q}_i(t)$ stetig und differenzierbar sind – diese Annahme ist berechtigt, weil die Roboterachsen trägheitsbehaftet sind und keine Bahnsprünge zulassen – ergibt die zeitliche Ableitung

$$\frac{\partial \boldsymbol{s}}{\partial t} = \sum_{i=1}^{n} \frac{\partial \boldsymbol{s}}{\partial q_i} \frac{\partial q_i}{\partial t} = J \frac{\partial q_i}{\partial t}, \qquad \partial \boldsymbol{s} = J \partial \boldsymbol{q} \tag{4.148}$$

mit J als Jacobi-Matrix. Ist diese invertierbar, kann durch Linearisierung

$$\Delta \boldsymbol{q} \approx J^{-1} \Delta \boldsymbol{s}$$

eine Vorschrift zur Berechnung der Achsinkremente gewonnen werden, wenn $\Delta \boldsymbol{s}$ ausreichend klein gewählt wird.

Für die DDD-Struktur ergeben sich aus Gl. (4.147) die Inkremente

$$\begin{aligned}
\Delta\varphi_1 &= \frac{p_x}{r^2} \Delta p_y - \frac{p_y}{r^2} \Delta p_x \\
\Delta\varphi_2 &= \frac{\cos(\varphi_2 + \varphi_3)}{l_2 \sin \varphi_3} \Delta r + \frac{\sin(\varphi_2 + \varphi_3)}{l_2 \sin \varphi_3} \Delta p_z \\
\Delta\varphi_3 &= \frac{r}{l_2 + l_3 \sin \varphi_3} \Delta r + \frac{p_z}{l_2 + l_3 \sin \varphi_3} \Delta p_z; \qquad r = \sqrt{p_x^2 + p_y^2}.
\end{aligned} \tag{4.149}$$

Die benötigten Absolutwerte der Winkel φ_1, φ_2, φ_3 werden durch Integration bestimmt (Bild 4.70). Die Lösungsgleichungen sind für diese Achskonfiguration besonders einfach und führen zu relativ schnellen Algorithmen [4.32]. Nachteil dieses Verfahrens ist, daß die aus der Linearisierung resultierenden Fehler sowie die durch die begrenzte Darstellungsgenauigkeit der Operanden im Rechner auftretenden Rundungsfehler sich bei der Integration fortpflanzen.

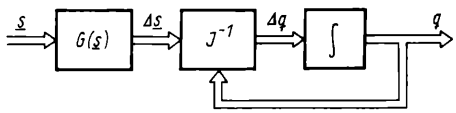


Bild 4.70. Inkrementelle Koordinatentransformation

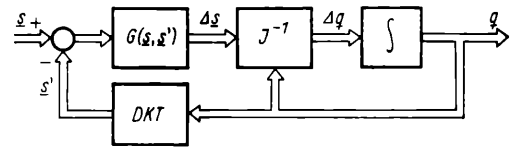


Bild 4.71. Inkrementelle Koordinatentransformation mit Rückführung

Zur Kompensation der Fortpflanzungsfehler kann eine Rückführung der berechneten Achspositionen gemäß Bild 4.71 durchgeführt werden. Es entsteht ein Servosystem, das dem Newtonschen Algorithmus zur Lösung nichtlinearer Gleichungssysteme äquivalent ist. Das Inkrement Δs wird nicht aus aufeinanderfolgenden Positionswerten bestimmt, sondern aus der Differenz von Sollwert und tatsächlich berechneten Koordinaten, die hierzu wieder in kartesische Werte transformiert werden. Vorteilhaft ist dabei, daß die DKT nicht innerhalb Δt_i ausgeführt werden muß, sondern nur nach einer bestimmten Anzahl von Iterationsschritten, die durch die Genauigkeitsforderung begrenzt wird, die Istposition zur Korrektur bereitstellt [4.33]. Durch zweckmäßige Wahl der Inkrementierung $G(s, s')$ können die durch sprunghafte Eingangsfunktionen $s - s'$ auftretenden Stabilitätsprobleme vermieden werden.

4.6.6. Realisierung der Koordinatentransformation

Genauigkeit und Taktzeit der inversen Koordinatentransformation sind aus der geforderten Positioniergenauigkeit und der Ersatzzeitkonstanten des kinematischen Systems zu bestimmen. Dies führt in der Regel zu der Forderung, einen vollständigen Koordinatensatz mit etwa 16 bit Auflösung in 20 bis 200 ms bereitzustellen. Aus der Rechentechnik sind zur Realisierung dieser Echtzeitforderungen zwei unterschiedliche Methoden bekannt. Die erste beruht auf der Off-line-Berechnung aller Funktionswerte und deren Abspeicherung in einer Tabelle. Die Rechenzeit wird dann nur durch die Tabellenzugriffszeit bestimmt. Bei der zweiten Methode werden die Funktionswerte durch geeignete Approximationsverfahren mit der gewünschten Genauigkeit berechnet. Ist der notwendige Wertevorrat einer Tabelle sehr groß, wird häufig eine Kombination beider Methoden genutzt, d.h., es wird eine Tabelle von Stützwerten der gesuchten Funktion bzw. Teile derselben angelegt und zwischen den Stützwerten wird die Funktion durch Interpolation approximiert. Entscheidend für den einzugehenden Kompromiß zwischen Rechenzeit und Speicherplatz ist dabei die zur Verfügung stehende Hardware. Erfahrungsgemäß ist für den gleichen Algorithmus das Produkt von Rechenzeit und Speicherplatzbedarf auf einem Rechner annähernd konstant, d.h., je mehr Funktionswerte als Tabelle abgelegt werden, desto kürzer wird die Rechenzeit und umgekehrt.

Zur Durchführung der Koordinatentransformation sind die Berechnungen der transzendenten Funktionen die aufwendigsten Operationen.

Eine Wertetabelle für trigonometrische Funktionen kann sowohl mit konstanter Inter-

vallbreite (lineare Tabelle) oder als nichtlineare Tabelle mit konstantem Interpolationsfehler angelegt werden. Umfang und Genauigkeit der Tabelle sind so zu wählen, daß der durch Akkumulation aller Darstellungs- und Rundungsfehler in der vollständigen Bestimmungsgleichung der Koordinate auftretende Gesamtfehler den vorgegebenen Wert nicht überschreitet.

Für eine Kosinustabelle mit 500 Stützstellen und 9stelliger Darstellungsgenauigkeit (31 bit) ergibt sich bei linearer Interpolation zwischen den Stützstellen gemäß

$$\cos\varphi = \frac{(\varphi - \varphi_n)(\cos\varphi_{n+1} - \cos\varphi_n)}{\varphi_{n+1} - \varphi_n} + \cos\varphi_n \tag{4.150}$$

im oberen Wertebereich (maximale Krümmung; s. Bild 4.72) ein maximaler Approximationsfehler von $\varepsilon \approx 10^{-6}$. Diese Genauigkeit ist für die Rundung auf ein 16-bit-Wort ausreichend. Addition der Rundungsfehler ergibt für die berechneten Koordinatenwerte eine erreichbare Genauigkeit von etwa 10^{-4}.

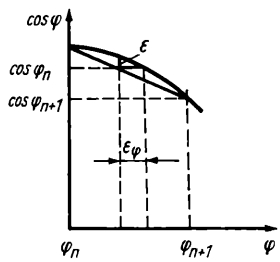


Bild 4.72
Approximationsfehler bei der linearen Interpolation zwischen Tabellenstützwerten einer Kosinustabelle

Da die trigonometrischen Funktionswerte bis auf das Vorzeichen in allen Quadranten gleich sind, werden Tabellen nur für einen Quadranten angelegt.

Grundsätzlich geeignet zur Approximation trigonometrischer Funktionen sind die Potenzreihe, die Kettenbruchapproximation, die rationale Approximation und die Tschebyschew-Reihe [4.34].

Die Potenzreihenentwicklungen, z. B.

$$\cos\varphi = \sum_{n=0}^{N} (-1)^n \frac{\varphi^{2n}}{(2_n!)} + \varepsilon(N), \tag{4.151}$$

approximieren zwar mit der geringsten Genauigkeit – für $N = 6$ wird $\varepsilon = 3 \cdot 10^{-5}$ –, sind aber einfach zu realisieren, da nur N Multiplikationen erforderlich sind.

Eine weitere Klasse von Verfahren kombiniert das Tabellenverfahren mit der Funktionsapproximation. Die Funktion wird in einem Intervall (f_i, f_{i+1}) durch Polynome möglichst niedriger Ordnung interpoliert; die Intervallbreite ist durch den maximalen Approximationsfehler gegeben [4.35].

Da in der digitalen Rechentechnik die Realisierung von Multiplikationen und Divisionen am aufwendigsten ist, sind zahlreiche Verfahren entwickelt worden, die arithmetische Ausdrücke durch die rechentechnisch schnell realisierbaren Operationen Addition, Subtraktion und Verschiebung iterativ berechnen [4.36] [4.37] [4.38].

Ein für die Koordinatentransformation geeignetes Verfahren stellt der CORDIC-Algorithmus dar (*co*ordinate *r*otation *di*gital *c*omputer). Mit ihm lassen sich die Funktionen

- Multiplikation einer Zahl mit sin φ, cos φ
- Berechnung von Wurzeln
- Berechnung von arctan

durch sukzessive Lösung von Differenzengleichungen bestimmen. Er ist besonders dann geeignet, wenn zur Berechnung dieser Funktionen keine schnelle Arithmetik zur Verfügung steht.

Das Prinzip und die Anwendung zur Koordinatentransformation werden in [4.39] ausführlich dargestellt.

4.6.7. Trajektorienplanung und Interpolation

Die Art der Bewegungsbahn des Endeffektors hängt von der vom Manipulator durchzuführenden Aufgabe ab. Für viele Anwendungen läßt sich die Steuerung des Effektors auf Geradenstücken vorteilhaft einsetzen, da die Anzahl der beim indirekten Teach-in anzufahrenden Punkte erheblich reduziert werden kann. Vom Bediener werden Anfangs- und Endpunkt der Geraden für die Position sowie die Geschwindigkeit v vorgegeben. Der Interpolator hat die Aufgabe, eine ausreichende Zahl von Zwischenpunkten zu erzeugen, die nach der IKT als Sollwertfolge an die Achsen ausgegeben werden. Sind P_A der Anfangs- und P_E der Endpunkt des Geradenstücks, ergeben sich für die Bahnlänge

$$l = \sqrt{(p_{xE} - p_{xA})^2 + (p_{yE} - p_{yA})^2 + (p_{zE} - p_{zA})^2} \tag{4.152}$$

und für die Interpolationsschrittweite

$$\Delta l = v\Delta t.$$

Die Interpolationstaktzeit Δt ist der Taktzeit der IKT gleichzusetzen. Ist diese größer als die kleinste Ersatzzeitkonstante der zu steuernden Antriebe, ist eine weitere Interpolation zwischen den berechneten Sollwerten der Achspositionen notwendig.

Durch Integration der Differenzengleichungen für die Bahninkremente läßt sich

$$\begin{aligned} \Delta l_x &= (p_{xE} - p_{xA})\frac{\Delta l}{l} \\ \Delta l_y &= (p_{yE} - p_{yA})\frac{\Delta l}{l} \\ \Delta l_z &= (p_{zE} - p_{zA})\frac{\Delta l}{l} \end{aligned} \tag{4.153}$$

die Differenz zwischen erreichtem und vorgegebenem Endpunkt zur Fehlerkompensation benutzen. Entsprechendes gilt für die Orientierung.

Mit Hilfe von Geradenstücken können einfache Weg-Zeit-Verläufe approximiert werden (Bild 4.73a). Bei schnellen Bewegungen treten dabei an den Zwischenpunkten Stellkraftsprünge der Antriebe auf, die zu Schwingungen des Roboterarms führen können. In der Regel ist das exakte Einhalten der Zwischenpunkte bei größeren Geschwindigkeiten aber nicht notwendig, so daß ein geeignetes Übergangsverhalten zwischen den Bahnsegmenten zulässig ist, das diese Schwingungen vermeidet. Eine konstante Beschleunigung innerhalb der Übergangsphase des Weg-Zeit-Verlaufs führt zur quadratischen Interpolation. Gemäß Bild 4.73b gilt

$$\Delta v_x = \dot{p}_{x2} - \dot{p}_{x1} = 2T\ddot{p}_x, \tag{4.154}$$

d.h., die Geschwindigkeit wird linear geändert und p_x verändert sich quadratisch. Diese Interpolation ist für alle Koordinaten von Position und Orientierung durchzuführen.

Zur Berechnung der Zwischenwerte werden damit neben P_0 und P_1 auch P_2 und die Geschwindigkeit des zweiten Bahnsegments benötigt. Der Übergang zur quadratischen Interpolation erfolgt zur Zeit $t = t_1 - T$. Die Bewegung des Effektors auf einer vorgegebenen Kreisbahn wird als zirkulare Interpolation bezeichnet.

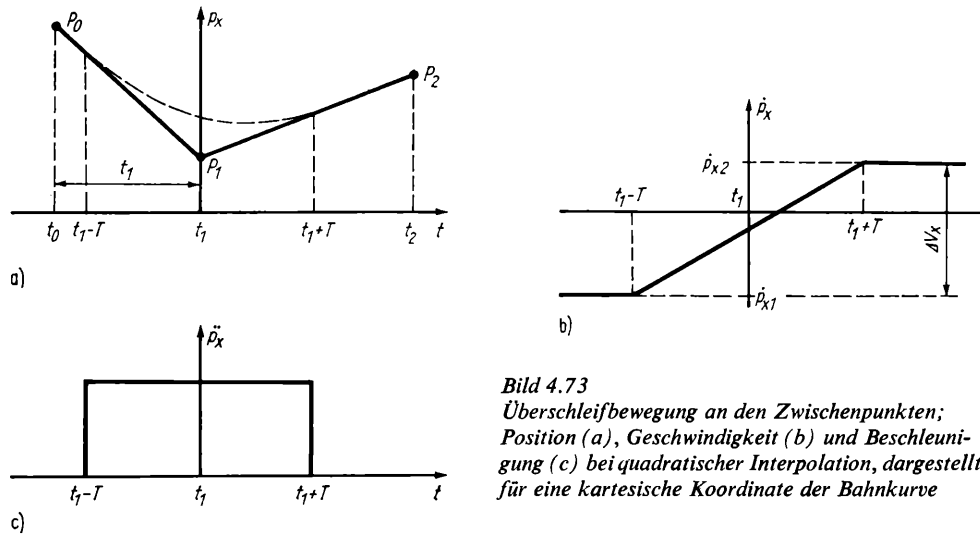


Bild 4.73
Überschleifbewegung an den Zwischenpunkten; Position (a), Geschwindigkeit (b) und Beschleunigung (c) bei quadratischer Interpolation, dargestellt für eine kartesische Koordinate der Bahnkurve

Ein einfaches Verfahren entsteht, wenn der Kreis durch Sehnen approximiert wird. Hierzu ist aus den Kreisstützpunkten zunächst der Radius R_0 zu bestimmen. Die Sehnenlänge

$$\Delta l = 2R_0 \sin \frac{\gamma}{2}; \qquad \gamma = 2 \arcsin \left(\frac{\Delta l}{2R_0} \right)$$

ist so zu wählen, daß die maximale Abweichung von der Kreisbahn eine vorgegebene obere Schranke nicht überschreitet.

Neben der Programmierung der Orientierung über indirektes Teach-in wird häufig auch die Orientierung der Positionsänderung unmittelbar nachgeführt, wenn sie mit der Bahnkurve in einem funktionellen Zuasmmenhang steht. Bei einer Kreisbewegung können beispielsweise die Winkel, die die Gerade durch P und R mit der durch den Kreis definierten Ebene bildet, eingegeben und bei der Bewegung nachgeführt werden.

Bei Bahnsteuerungen sind Beschleunigungs- und Bremsphasen so zu gestalten, daß die technologischen Bedingungen des Prozesses eingehalten werden. Die Beschleunigungsvorgabe ist außerdem von den konstruktiven Merkmalen des Manipulators abhängig.

Zur Beschreibung komplizierter Raumkurven lassen sich Polynome höherer Ordnung einsetzen. Sind n Bahnpunkte $r(t_i)$ und die Geschwindigkeiten $\dot{r}(t_i)$ gegeben, können die Bahnzwischenpunkte aus

$$r(t) = \sum_{i=0}^{n-1} a_i t^{b_i} + c_i \tag{4.155}$$

ermittelt werden. Die Koeffizienten a_i, b_i und c_i sind durch die Anfangs- und Endbedingungen sowie Randbedingungen, die sich durch zusätzliche Kriterien zum Zeitverhalten oder zu den Bahnabweichungen ergeben, zu berechnen.

Wegen des hohen Rechenaufwands sind solche Interpolationsverfahren für große n nur off-line durchführbar.

Reduziert man den Polynomgrad auf einige benachbarte Bahnpunkte, läßt sich die Interpolation sukzessive auf die gesamte Trajektorie anwenden.

Zur Bewegungsplanung von Bahndaten, die über äußere Sensoren erfaßt werden, ist das Frame-Konzept geeignet [4.43]. Es beruht auf der Transformation von Koordinatensystemen in Matrixschreibweise.

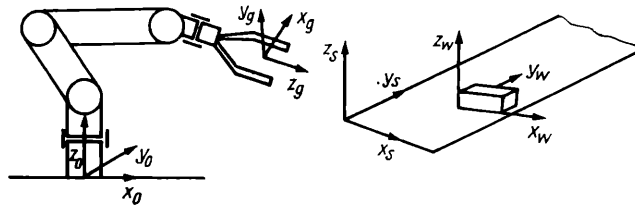


Bild 4.74
Koordinatensysteme zur Bewegungsplanung

Wird ein Werkstück gemäß Bild 4.74 durch eine Frame (Koordinatensystem) T_w beschrieben, in dem die Greifposition festgelegt ist, gilt für seine Darstellung im Basissystem x_0, y_0, z_0:

$$T_w^0 = T_s^0 T_w^s; \tag{4.156}$$

T_s^0 homogene Transformation, die das Koordinatensystem x_s, y_s, z_s in Basiskoordinaten beschreibt
T_w^s beschreibt objekteigene Koordinatensystem in Stationskoordinaten x_s, y_s, z_s,
T_w^s kann z.B. durch die Erkennung von Lage und Orientierung eines Werkstücks mittels visueller Erkennung bestimmt werden;
x_s, y_s, z_s können dabei i.allg. auch ein bewegtes Koordinatensystem darstellen (Transportband).

Das Frame T_g beschreibt die Effektorstellung im Basissystem. Die Transformation T_x^g beschreibt das werkstückeigene Koordinatensystem im greifereigenen Koordinatensystem.

Mit Hilfe der Beziehungen

$$T_g = T_s^0 T_w^s T_x^g \tag{4.157}$$

lassen sich die Bahndaten berechnen, um aus der momentanen Greiferstellung eine in T_w^s vorgegebene Greifposition zu erreichen. Die Methode ist auch für die Arbeit mit bewegten Objektkoordinaten geeignet; aus Gl. (4.157) ergeben sich in diesem Fall Daten für eine Folgebewegung. Die zeitlichen Änderungen der Koordinaten werden über geeignete äußere Sensoren (Geschwinidgkeit des Transportbands oder Bewegungsdaten des Objekts) erfaßt; damit wird $T_w^s = T_w^s(t_i)$.

Die Angabe von Position und Orientierung durch eine Frame wird in zahlreichen Programmiersprachen benutzt. Dabei ist zu unterscheiden, ob die Bewegungsdaten durch Angabe der Greiferstellung oder durch Angabe der Objektzustände programmiert werden (s. Abschn. 6.). Intern werden immer die notwendigen Positions- und Orientierungsdaten des Effektors durch Lösen der sich aus den Matrixmultiplikationen ergebenden Gleichungssysteme berechnet. Dabei sind die durch die Kinematik des konkreten Industrieroboters bedingten Beschränkungen für die Greiferstellung zu beachten. Durch günstige Wahl der Koordinatensysteme, insbesondere der Orientierungsangaben, läßt sich der Rechenaufwand reduzieren, d.h., die zu den Frames gehörigen Matrizen müssen möglichst viele Nullelemente enthalten.

5. Aktionssteuerung

5.1. Grundfunktionen und Steuerungsentwurf

5.1.1. Grundfunktionen

Die Aktionssteuerung hat die Aufgabe, die Aktionsbefehle des Roboterprogramms auszuführen, und stellt im Prinzip eine binäre Steuerung für die Roboterperipherie dar. Sind in der Peripherie zusätzlich analoge Steuerungsfunktionen (z. B. Schweißstromregelungen) zu realisieren, so ist dafür eine entsprechende Funktionsgruppe zwischen Bewegungssteuerung und Aktionssteuerung einzuordnen. Im folgenden wird darauf nicht näher eingegangen.

Die Aktionssteuerung entspricht im Prinzip der PC (*p*rogrammable *c*ontrol) einer Werkzeugmaschinensteuerung. Im einfachsten Fall wird durch sie die kombinatorische Verknüpfung von binären Prozeßsignalen (z. B. Endlagenschalter-, Lichtschranken- und Initiatorsignale) gebildet und dadurch die Ansteuersignale für binäre Stellglieder (z. B. Schaltschütze, Magnetventile, Einfachantriebe) erzeugt. Im allgemeinen Fall realisiert die Aktionssteuerung mehrere voneinander unabhängige sequentielle Teilautomaten.

Die Grundstruktur des Teilautomaten ist im Bild 5.1 dargestellt. Der Automat erhält über die Ausgangsgrößen $\underline{x}$ des Steuerungsobjekts die notwendigen Informationen, um dessen Zustand $\underline{q}$ zu ermitteln. Aus den Eingangsgrößen $\underline{x}_w$, die sich aus dem entsprechenden Aktionsbefehl herleiten, den Ausgangsgrößen $\underline{x}$ des Steuerungsobjekts und ihrem inneren Zustand $\underline{z}$ bildet die Steuerung entsprechend dem Steueralgorithmus die Stellgrößen $\underline{y}$ und den Folgezustand $\underline{z}$. Zur Beschreibung des durch den Automaten zu realisierenden Ablaufs sind möglich

- Automatengleichungen
- Graphen und Netze
- Sprachen.

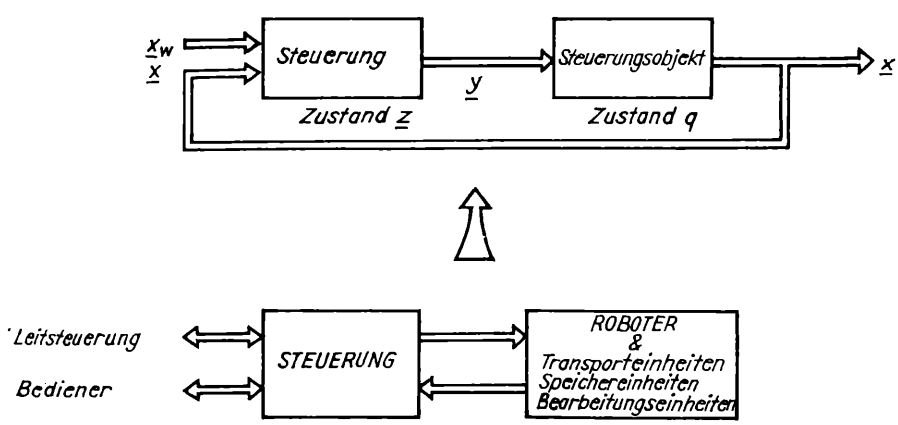


Bild 5.1. Grundstruktur eines Steuerungssystems

Für den Entwurf und zur Simulation des Zusammenwirkens von Steuerung und Steuerungsobjekt ist es sinnvoll, auch für das Steuerungsobjekt das gleiche Beschreibungsmittel zu wählen. Werden dafür die Automatengleichungen benutzt, so sind die Zustandsgleichungen für die Steuerung

$$z' = f(z, x, x_w), \tag{5.1}$$

das Steuerungsobjekt

$$q' = \varphi(q, y) \tag{5.2}$$

und die entsprechenden Ausgangsgleichungen

$$y = g(z, x, x_w) \tag{5.3}$$

$$x = \gamma(q, y) \tag{5.4}$$

anzusetzen. Die Gln. (5.1)bis (5.4) geben zusammengefaßt die Zustands- und Ausgangsgleichungen für den Gesamtautomaten (Steuerung und Steuerungsobjekt). Ähnliches ergibt sich bei der Verwendung von Netzen. Das Gesamtnetz setzt sich aus den Teilnetzen für die Steuerung und das Steuerungsobjekt zusammen.

Die Ermittelung der Struktur der entsprechenden Graphen und Netze oder der Funktionen in den Gln. (5.1) bis (5.4) erfogt nach einer zweckmäßigen Strategie, die in den nachfolgenden Abschnitten behandelt wird.

Da es sich sowohl bei der Entwicklung der Aktionssteuerung als auch beim logischen Entwurf von Roboterprogrammen im Prinzip um die Behandlung binär gesteuerter Abläufe handelt, ist diese Strategie in beiden Fällen (PC-Entwurf und technologische Ablaufplanung) in fast gleicher Weise verwendbar.

Sie haben darüber hinaus noch Bedeutung zur Gestaltung des Teilsystems zur Koordinierung des steuerungsinternen Ablaufs (Echtzeit-Multitask-System).

5.1.2. Entwurfsstrategie

Die Strategie für den Entwurf von Steuerungssystemen für diskrete technische Prozesse beruht auf der im Bild 5.1 dargestellten Grundstruktur des Steuerungssystems. Sie geht davon aus, daß der Entwurfsprozeß sich iterativ und in verschiedenen relativ abgeschlossenen Phasen,

- der Modellierungsphase,
- der Entwurfsphase und
- der Realisierungsphase,

[5.2] vollzieht.

Zu Beginn des Entwurfsprozesses liegen dem Entwerfer

- Informationen über das Steuerungsobjekt und die
- Steuerungsaufgabe

vor. Die Steuerungsaufgabe legt die zu realisierenden Funktionen auf dem Steuerungsobjekt – Bewegungsabläufe, Betriebsarten, Verriegelungen u.a. – fest. Der Gewinnung und fortlaufenden Präzisierung der Informationen über den zu steuernden Prozeß dient eine

Prozeßanalyse [5.1]. Sie hat den Charakter einer theoretischen Prozeßanalyse und führt zu einem Modell des Steuerungsobjekts, das auf der Basis des

- Verhaltens von elementaren Objekten und der
- Relation zwischen den elementaren Objekten

ermittelt wird.

An die Modellierungsphase schließt sich die Entwurfsphase und die Realisierungsphase an.

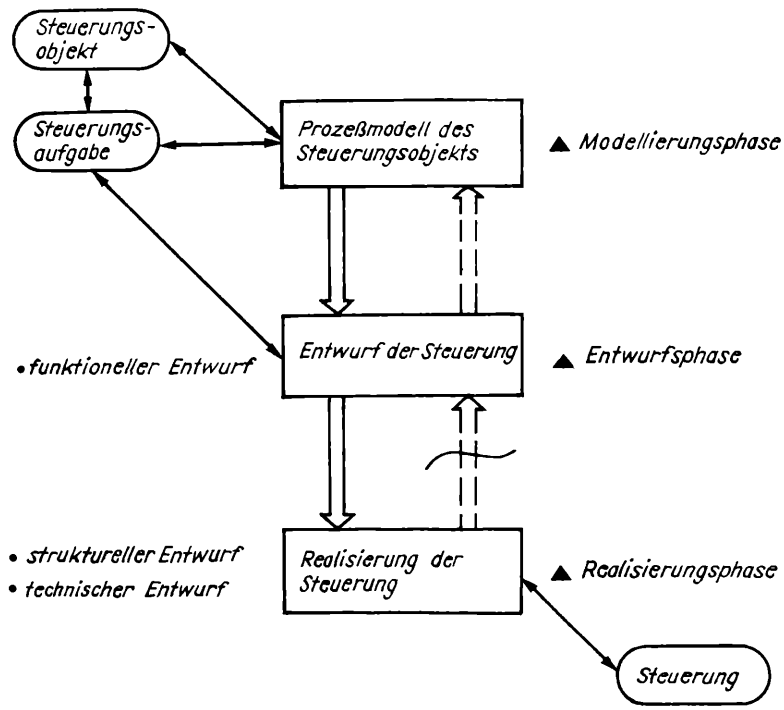


Bild 5.2. Phasen des Steuerungsentwurfs

Der Unterteilung in eine Entwurfsphase und Realisierungsphase liegt das Bestreben zugrunde, die Steuerung zuerst auf einem strategischen und algorithmischen Niveau zu entwerfen und dann diesen Steuerungsalgorithmus mit der speziell ausgewählten Realisierungstechnik praktisch umzusetzen.

Der in der Entwurfsphase entstehende funktionelle Entwurf ist das Ergebnis eines Iterationsprozesses, in dessen Verlauf die Unschärfe der Formulierung der Steuerungsaufgabe abnimmt. Der funktionelle Entwurf liefert die Beschreibung der Steuerung mit einem vorgegebenen Beschreibungsmittel, das sowohl für die Fachleute der Technologie als auch für die des Steuerungssystems in gleicher Weise verständlich sein muß. Die ingenieurtechnische Anschaulichkeit läßt dabei grafische Beschreibungsmittel als besonders geeignet erscheinen. Ziel des funktionellen Entwurfs ist es, unabhängig von der Form der technischen Realisierung – Hardware oder Software –, eine Beschreibung einer Steuerung vorzulegen, die verschiedensten Fachleuten zur Untersuchung auf Vollständigkeit, Widerspruchsfreiheit und Eindeutigkeit dient.

In der Realisierungsphase oder Implementierungsphase vollzieht sich die Umsetzung des funktionellen Entwurfs in Hardwarekomponenten oder Softwarekomponenten des zu konfigurierenden Steuerungssystems.

Jeder Entwurf eines Steuerungssystems vollzieht sich gedanklich in diesen drei Phasen. Werden die Phasen der Prozeßanalyse und des funktionellen Entwurfs gedanklich vom Entwerfer vorweggenommen und dieses Ergebnis unmittelbar in der Realisierungsphase als Hardware oder Software niedergelegt, so spricht man vom *intuitiven* Entwurf. Werden die Phasen jedoch getrennt abgearbeitet und dokumentiert, so spricht man vom *systematischen* Entwurf. Die letzte Vorgehensweise bietet folgende Vorteile:

- Formalisierbarkeit der Arbeitsschritte der Realisierungsphase,
- Dokumentation der Steuerungsfunktionen auf dem Niveau des Beschreibungsmittels,
- Portabilität des Ergebnisses des funktionellen Entwurfs,
- Anwendung mathematischer Theorien bei der Strukturierung und der Analyse auf Vollständigkeit und Widerspruchsfreiheit im funktionellen Entwurf,
- Aufteilung der Steuerungsfunktionen im Sinne einer Hierarchie.

Während die Phase des funktionellen Entwurfs schöpferischen Charakter trägt, ist die Realisierungsphase weitestgehend formalisierbar. Damit ist die Trennung in Entwurfsphase und Realisierungsphase eine Abtrennung der formalisierbaren Anteile des Entwurfs. Sowohl für die Entwurfsphase als auch für die Realisierungsphase ist abgesehen von Steuerungsaufgaben geringen Umfangs ein CAD-System auf einem komfortablen Arbeitsplatzrechner notwendig.

5.1.3. Steuerungsrealisierung

An den funktionellen Entwurf schließt sich die Realisierung der Steuerung an. In dieser Phase erfolgt die Umsetzung des funktionellen Entwurfs auf der vorgegebenen Realisierungsbasis. Diese Phase kann für sich noch aufgeteilt werden in einen

- *strukturellen Entwurf* – Umsetzung des funktionellen Entwurfs auf das von der Realisierungsbasis geforderte Beschreibungsniveau –
  und einen
- *technischen Entwurf* – Konfigurierung der Realisierungsbasis und Fixierung der technischen Randbedingungen.
  Es wird zwischen einer
- *verbindungsprogrammierten Realisierung* mit Logik- und Speicherbausteinen
  und einer
- *speicherprogrammierten Realisierung* mit Bitprozessoren und 4-Bit-, 8-Bit- bzw. 16-Bit-Mikroprozessoren (z.B. U880, diskreter Bitprozessor)

unterschieden. Der Zuschnitt der Steuerung auf den betreffenden Anwendungsfall wird bei der verbindungsprogrammierten Technik durch Festlegung – *Programmierung* – der *Verbindungsstruktur* zwischen den Steuerungsbauelementen erreicht und bei der speicherprogrammierten Technik durch Konfigurierung der zur Auswahl stehenden Hardware, z.B. E/A-, Speicherbaugruppen u.a., und Software und durch *Programmierung des Speichers* für das Anwenderprogramm.

Der strukturelle Entwurf muß folglich bei den verbindungsprogrammierten Steuerungen die Ansteuergleichungen für die Logik- und Speicherbausteine liefern und bei den speicherprogrammierten Steuerungen das Anwenderprogramm in der für die Steuerung festgelegten Sprache ergeben.

Im einzelnen sind dann bei der Umsetzung des funktionellen Entwurfs die im Bild 5.3 angegebenen Schritte zu gehen.

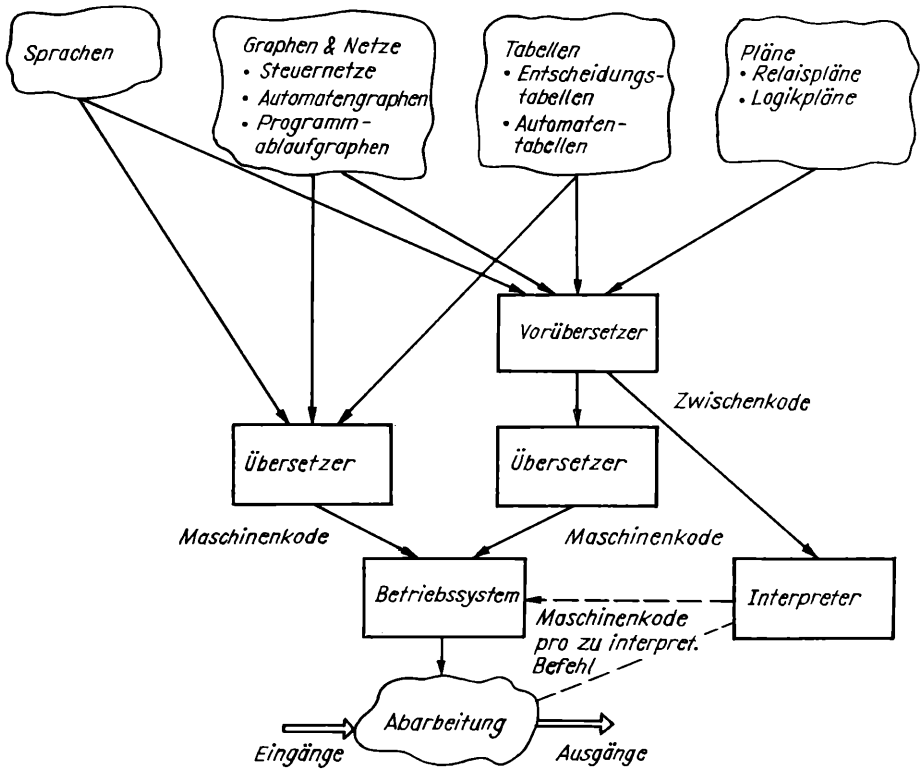


Bild 5.3. Schritte in der Realisierungsphase

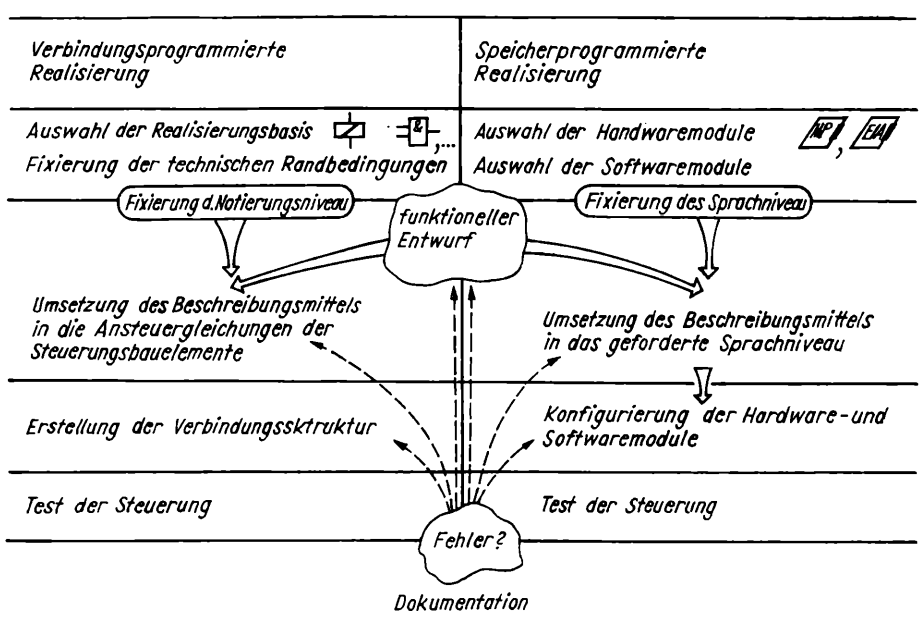


Bild 5.4. Niveaus in der Umsetzung des Beschreibungsmittels

Tafel 5.1. Auszug aus dem DIN-Entwurf 19239
a) binäre Operationen in Funktionsplänen und Relaisplänen; b) digitale Operationen in Funktionsplänen

Tafel 5.1a

Benennung	Mnemotechn. Zeichen		Symbolik	
	deutsch	englisch	Funktionsplan	Relaisplan
UND	U	A	E1 E2 & A1	E1 E2 A1
ODER	O	O	E1 E2 >=1 A1	E1 E2 A1
Negation (NICHT)	N	N	am Eingang E1 am Ausgang A1	E1 M1 A1
Antivalenz (Exklusiv-ODER)	XO	XO	E1 E2 =1 A1	E1 E2 A1
Zuweisung	=	=	A1	A1
Setzen	S	S	S	(S)
Rücksetzen	R	R	R	(R)
Klammern	( )	( )	durch Zusammenwirken der Elemente (Strompfade) realisiert	
Zählen, vorwärts	ZV	CU	Z +	
Zählen, rückwärts	ZR	CD	Z −	
Impuls	I	P	T 1	
Ein-/Ausschalt-verzögerung	E A	R F	T T 0 T 0 T	

Tafel 5.1 (Fortsetzung)

Tafel 5.1 b

Benennung	Mnemotechn. Zeichen deutsch	Mnemotechn. Zeichen englisch	Symbolik
Addieren	ADD	ADD	+
Subtrahieren	SUB	SUB	−
Multiplizieren	MUL	MUL	×
Dividieren	DIV	DIV	:
Größer, größer gleich	GR GRG	GT GTE	> >=
Gleich	GL	EQ	=?
Kleiner, kleiner gleich	KL KLG	LT LTE	< <=
Kodieren dezimal/dual	DED	DEB	DE DU
Kodieren dual/dezimal	DUD	B/D	DU DE

5.2. Mittel zur Funktionsbeschreibung

5.2.1. Überblick

Zur Beschreibung der Funktion von Steuerungssystemen zu Steuerung diskreter technischer Prozesse haben sich im Laufe der Entwicklung eine Vielzahl von Mitteln herausgebildet. Diese Beschreibungsmittel waren vielfach an eine neu einzuführende Realisierungsbasis gebunden und haben sich bis heute gehalten. Folgende Mittel sind üblich:

Diagramme

In Weg-Schritt-, Weg–Zeit- oder Taktdiagrammen wird der von der Steuerung durchzuführende Ablauf beschrieben. Mit Wirkungspfeilen wird die Abhängigkeit zwischen den verschiedenen Zeitabläufen charakterisiert.

Tabellen

Die bekanntesten Tabellendarstellungen sind die

- Automatentabelle
- Entscheidungstabelle.

Aus diesen Tabellen ist abzulesen, wie sich nach Eintreten einer neuen Prozeßsituation der Zustandsübergang – Übergang vom Istzustand zum Nachfolgezustand – vollzieht und welche Aktionen auszuführen sind.

Sprachen (s. Abschn. 6.).

Schaltpläne und Funktionspläne

Die Beschreibung der Steuerungsfunktion mit Schaltplänen als

- Relaisplan
- Logikplan

hat sich durch die in der Vergangenheit übliche Realisierung von Steuerungen mit Relais oder anderen logischen Bauelementen eingebürgert. Daher wird diese Notierungsform auch heute noch für speicherprogrammierte Steuerungen verwendet, obwohl die Realisierungstechnik eine ganz andere ist. Eine Verallgemeinerung dieser Herangehensweise findet sich in der Methode der

- Funktionspläne (Tafel 5.1, DIN 19239) wieder.

Programmablaufpläne (s. TGL 22451)

Graphen und Netze

Die auf der Graphen- bzw. Netztheorie beruhenden Beschreibungsmittel sind ablauforientiert. Die gebräuchlichsten sind

- Automatengraph
- Programmablaufgraph (s. Abschn. 5.2.2.)
- Stellen-Transitions-Netze (s. Abschn. 5.2.3.)
- Prädikat-Transitions-Netze.

Mit wachsender Leistungsfähigkeit der speicherprogrammierten Steuerungen setzen sich die *ablauforientierten* Beschreibungsmittel auf der Basis von Graphen und Netzen im Zusammenhang mit einer Sprache immer mehr durch.

5.2.2. Programmablaufgraphen

Der Programmablaufgraph (PAG) ist aus den in der Rechentechnik zur Notierung von Algorithmen bekannten Programmablaufplänen hervorgegangen. In der Darstellung lehnt sich der PAG an die Sinnbildmethode (TGL 22451) an. Die Aktionen oder Operationen werden grafisch durch ein Rechteck dargestellt und mit den durchzuführenden Operationen und den zugeordneten Ausgangsgrößen beschriftet. Die binären Entscheidungen des Programmablaufplans, dargestellt durch eine Raute, werden im PAG als

binäre Funktion der Eingangssignale der Steuerung interpretiert. Das Element hat daher zwei Ausgänge: einen, bei dem der Funktionswert 0, und einen, bei dem er 1 ist. Bei der Konstruktion des PAG sind folgende Eigenschaften zu beachten:

- Jede Operation liegt in dem Graph auf einem geschlossenen Weg, der keine weitere Operation enthält:
  - Existenz einer Eigenschleife für die Operation.
- In jedem Programmablaufgraph wird nur immer eine Operation ausgeführt:
  - Eigenschaft einer Zustandsmaschine.
- Die von einer Operation wegführenden Wege müssen wieder zu einer Operation führen und hinsichtlich ihrer logischen Bewertung paarweise disjunkt sein:
  - Widerspruchsfreiheit und Vollständigkeit.

Ausführliche Erläuterungen findet der interessierte Leser in [5.21]. Ein Beispiel soll die Anwendung des Beschreibungsmittels verdeutlichen. Für den Bewegungsablauf einer linearen Einzelachse (Bild 5.5) ist vorgegeben, daß die Bewegung mit $t = 1$ gestartet wird und so lange anhält, wie $t = 1$ ist. Die Bewegung wird bei $t = 0$ abgebrochen (Halt). Bei erneutem Start der Bewegung wird die Bewegungsrichtung beibehalten. An den Endlagenschaltern E1 und E2 erfolgt die Umkehr der Bewegungsrichtung. Beim Einschalten der Steuerung wird der Rechtslauf eingenommen.

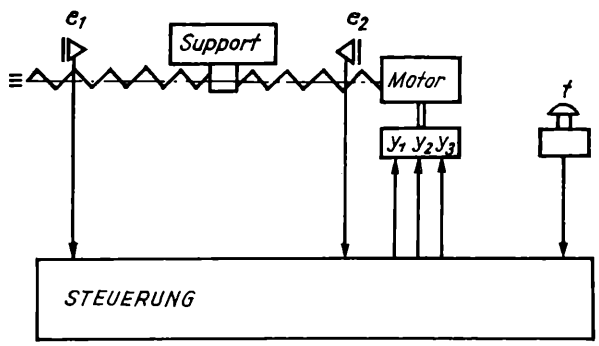
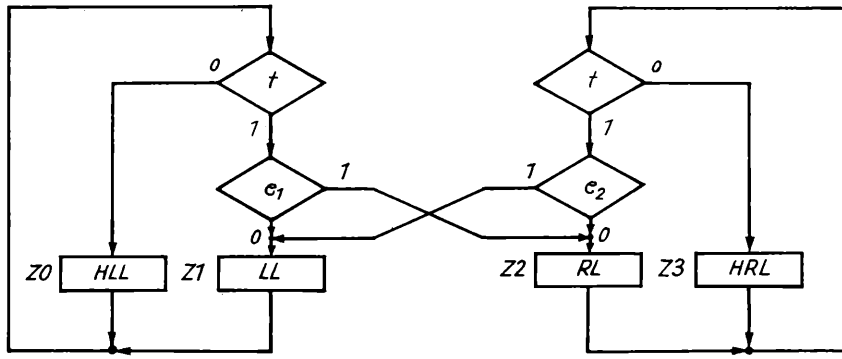


Bild 5.5
Technologisches Modell für eine translatorische Achse

Zwei Varianten des Entwurfs sind im Bild 5.6a und b dargestellt. In der ersten (Bild5.6a) werden alle möglichen Zustände dargestellt, während in der zweiten die Zustände nur das Gedächtnis für die eingenommene Bewegungsrichtung widerspiegelt. Das Starten und Anhalten wird über die Ausgangsgrößen durch logische Verknüpfungen von t mit den Zuständen erreicht.

5.2.3. Netze

Die Verfahren auf der Grundlage von PAG und Automatengraphen orientieren darauf, daß die Zustandsänderungen des Gesamtsystems betrachtet werden. Diese Vorgehensweise hat aber den wesentlichen Nachteil, daß die Übersicht über die kausalen Zusammenhänge von Zustandsübergängen innerhalb des zu steuernden Systems verlorengeht, d.h., man kann aus den unter dem Aspekt der Zusammenfassung zusätzlich zu den sequentiellen Prozessen entstandenen Sequenzen nicht mehr diejenigen erkennen, die tatsächlich in einem ursächlichen Zusammenhang stehen. Außerdem besteht ja das Ziel einer Prozeßbeschreibung letztlich darin, diese in eine entsprechende Steuereinrichtung abzubilden bzw. umzusetzen. Bei einem hinreichend großen System, bei dem ein ein-

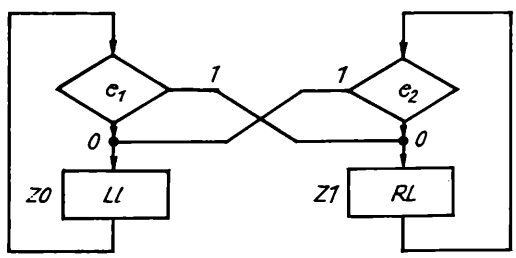


H Halt
LL Linkslauf
RL Rechtslauf

Zustand	Zustandsgrößen	Operation	Ausgangsgrößen
	$z_0\ z_1\ z_2\ z_3$		$y_1\ y_2\ y_3$
Z_0	1 0 0 0	HLL	0 0 0
Z_1	0 1 0 0	LL	1 0 1
Z_2	0 0 1 0	RL	0 1 1
Z_3	0 0 0 1	HRL	0 0 0
AUSGANGSTABELLE			

$y_1 = z_1$
$y_2 = z_2$
$y_3 = z_1 \vee z_2$

a)



Zustand	Zustandsgrößen	Operation	Ausgangsgrößen
	$z_0 z_1$		$y_1\ y_2\ y_3$
Z_0	1 0	LL	t 0 t
Z_1	0 1	RL	0 t t
AUSGANGSTABELLE			

$y_1 = t \cdot \bar{z}_1$
$y_2 = t \cdot z_1$
$y_3 = t$

b)

Bild 5.6. Beispiel der Beschreibung der Steuerung der translatorischen Achse mit PAG
a) Steuerung Variante 1; b) Steuerung Variante 2

zelner Bearbeiter nicht mehr in der Lage ist, alle systemrelevanten Vorgänge auf einmal zu überblicken, erscheint es mit vertretbarem Aufwand kaum möglich, ein für alle Teilkomponenten gemeinsames Zeitraster a priori herzustellen.

An einem kleinen Beispiel soll dieser Sachverhalt verdeutlicht werden. Bild 5.7a zeigt zwei Automatengraphen A und A', die jeweils eine Positionierungssteuerung einer Werkzeug-(Werkstück-)palette mit den drei Positionen *0*, *1* und *2* beschreiben sollen. Die jeweiligen Zustandsübergänge a_i bzw. b_i werden aus den Eingangsvariablen auf eine hier nicht näher zu beschreibende Art und Weise ermittelt; ihre Reihenfolge liegt jedoch in Form der beiden ableitbaren sequentiellen Prozesse $(a_1, a_2, a_3, a_1, a_2 \ldots)$ bzw. $(b_1, b_2, b_3, b_1, b_2 \ldots)$ fest.

Wären die in ihren Teilsystemen ablaufenden Prozesse völlig voneinander unabhängig, so könnte man das Gesamtsystem durch einen Zustandsgraphen mit neun Zuständen und allen entsprechenden Übergängen abbilden. Dieser Fall ist aber praktisch nicht von Interesse, da unverkoppelte nebenläufige Prozesse selbstverständlich auch unabhängig beschrieben und gesteuert werden können. Wesentlich problematischer wird die Beschreibung, wenn aus der Aufgabenstellung oder Prozeßbeobachtung Verkopplungen

ableitbar sind, etwa in der Art: Zustandsübergang b_1 in A' kann nur erfolgen, wenn die ihn implizierenden Eingangsvariablen im Zustand 0′ wahr sind und vorher der Zustandsübergang a_1 in A erfolgt ist, der Übergang a_3 hänge in gleicher Weise von b_2 ab.

Eine solche Verkopplungsbedingung könnte beispielsweise die Folge der Aktivitäten eines Beschickungs- oder Montageroboters sein, der mit beiden Paletten arbeitet. Im Bild 5.7b ist der Zustandsgraph des Gesamtsystems dargestellt, der durch Zusammenfassung gleichzeitig vorliegender Zustände der Teilautomaten A bzw. A' entsteht. Dabei wurde die topologische Anordnung der Teilprozesse beibehalten, so daß die Übergänge a_i wieder in horizontaler und die Übergänge b_i in vertikaler Richtung verlaufen. Interessant ist, daß zwei Hilfsvariablen (*3* bzw. *3′*) eingeführt werden mußten, um einen der Aufgabe gerechten und determinierten Automaten zu erhalten. Dies liegt eindeutig an der Verkopplungsbedingung, da jetzt z. B. der Zustand (*10′*) eingenommen werden kann, nachdem der Übergang b_1 noch nicht stattgefunden hat (*10′ 3*) oder nachdem der gesamte Teilprozeß (b_1, b_2, b_3) abgelaufen ist (*10′ 3′*). Dies gilt sinngemäß auch für die zusammengefaßten Zustände (*20′*) (*12′*) und (*22′*), die gemäß Bild 5.7b ebenfalls je in zwei Zustände aufgespaltet werden.

Im Bild 5.7b sieht man, daß das zeitliche Verhalten des horizontalen Prozesses darüber entscheidet, welchen der fünf möglichen Zustände der vertikale Prozeß nach dem Zustandsübergang a_2 einnimmt. Ziel einer Beschreibung kann aber nur ein determiniertes,

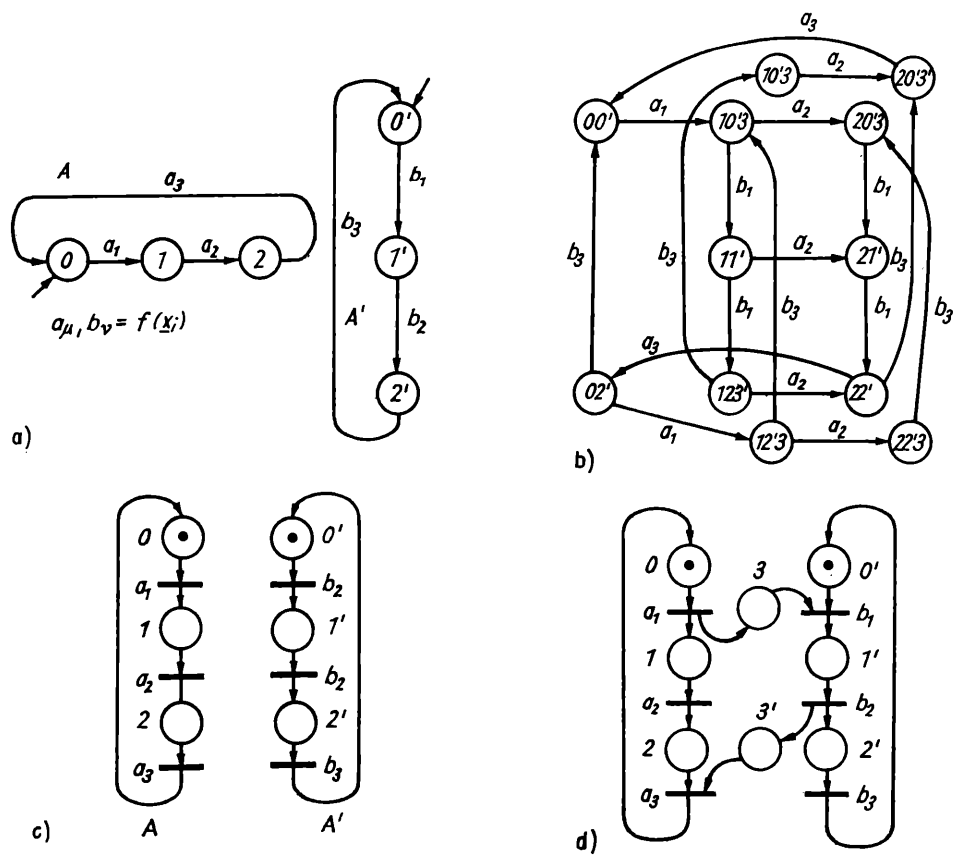


Bild 5.7. Gegenüberstellung der Beschreibung paralleler Abläufe

a) Darstellung der sequentiellen Teilabläufe mittels Automatengraphen (AG); b) Darstellung des verkoppelten Ablaufs mittels AG; c) äquivalente Darstellung von a) mit BE-Systemen; d) Darstellung des verkoppelten Gesamtablaufs mit BE-Systemen

also von zufälligen Zeitbedingungen freies und somit zeitinvariantes System sein, will man sich nicht der Arbeit einer Analyse aller denkbaren zeitlichen Varianten der Prozeßverläufe unterziehen. Dabei ist das Erkennen des Kausalgefüges, d.h. Ursache-Wirkung-Beziehung eines Systems nicht nur für den Fall parallel zueinander ablaufender Prozesse interessant, sondern es ermöglicht auch bei rein sequentiellen Problemstellungen eine große Variantenvielfalt für die Realisierung. So muß eine einmal willkürlich festgelegte Sequenz nicht immer die optimale Lösung sein; ihre Änderung ist aber nur dann möglich, wenn Aussagen darüber vorliegen, welche Elemente der gewählten Sequenz permutiert werden können und welche aus den Gegebenheiten des Steuerobjekts heraus tatsächlich in der angegebenen Reihenfolge ausgeführt werden müssen. Diese Überlegungen führten *C.A.Petri* zur Begründung seiner nach ihm benannten Netztheorie [5.9] [5.23], deren Grundlage die sog. Bedingung-Ereignis-Systeme (BE-Systeme) darstellen. Die angestrebte Ursache-Wirkung-Beziehung wird in den BE-Systemen ausgedrückt durch die Paare Ereignis–Bedingung bzw. Bedingung–Ereignis, und der Zustand des Systems wird durch die Angabe der jeweils geltenden Prozeßbedingungen charakterisiert, d.h. also durch eine Menge nebenläufiger und damit kausal unabhängiger Bedingungen. Zur Unterscheidung vom Zustandsbegriff in der Automatentheorie führte *Petri* den Begriff des Falls ein.

Die grafische Darstellung aller möglichen Fälle eines Systems heißt Fallgraph (B, E, F, R_F), wobei B die Menge der Bedingungen, E die Menge der Ereignisse und F die Menge der Fälle darstellt, die wiederum über die Relation R_F miteinander verknüpft sind. Für F und R_F gilt dabei

$$F \subseteq P(B) \tag{5.5}$$

und

$$R_F \subseteq F \times E \times F. \tag{5.6}$$

Ein Fall f wird also durch Eintreten eines Ereignisses e in einen anderen Fall f' überführt, was im Netz der verknüpften Bedingungen und Ereignisse bedeutet, daß e eintreten kann, wenn alle seine Vorbedingungen und keine seiner Nachbedingungen erfüllt sind. Man sagt: e hat Konzession. Durch das Eintreten von e wird die Existenz der Bedingungen von Fall f beendet, und es werden die Bedingungen von Fall f' erfüllt. Es soll dabei gelten, daß jedes Ereignis in mindestens einem Fall Konzession hat, sonst kann es nie eintreten und somit weggelassen werden. Der entstehende Fallgraph soll außerdem stark zusammenhängend sein, wodurch jeder Fall in jeden anderen Fall überführt werden kann. Damit sind Bedingungen und Ereignisse im Netz durch die Relation V mit

$$V \subseteq B \times E \cup E \times B \tag{5.7}$$

verknüpft. Als grafische Darstellung werden für die Bedingungen Kreise, für die Ereignisse Striche oder Kästchen, für die Verbindungen Pfeile und für die Fälle Punkte in den Kreisen angegeben (Bild 5.8a). Die Punkte in den Kreisen geben die Markierung des Systems an. Unter einer Markierung m wird die Abbildung

$$m: B \rightarrow \{0, 1\} \tag{5.8}$$

verstanden, und zwar so, daß für einen Fall f gilt

$$m_f(b) = \begin{cases} 1, & \text{falls } b \in f \\ 0 & \text{sonst.} \end{cases} \tag{5.9}$$

In der grafischen Darstellung bedeutet $m(b) = 1$, daß in dem zugehörigen Bedingungskreis eine Marke (dargestellt als Punkt) eingetragen ist.

Bereits mit diesen einfachen Grundbegriffen und Hilfsmitteln ist es möglich, das zu steuernde System nach Bild 5.7b wesentlich übersichtlicher und vor allem korrekt und invariant zu beschreiben. Bild 5.7c zeigt die beiden Prozesse und Bild 5.7d ihre Verkopplung. Die Bezeichnungen der Zustände bzw. Übergänge wurden dabei auf die Bedingungen bzw. Ereignisse übertragen. Dabei werden den Ereignissen die Zustandsübergänge und den Bedingungen die Zustände der Teilautomaten zugeordnet, während die Gesamtzustände von Bild 5.7b jeweils einem Fall (einer erreichbaren Markenkonstellation) des Netzes entsprechen. Ein Ereignis kann demnach eintreten, wenn seine Vorbedingung (je eine Marke in allen Vorplätzen in der Darstellung) und die Übergangsbedingung vom Automatengraphen (a_i bzw. b_i) erfüllt sind. Mit dieser Beschriftung wird für BE-Systeme bereits eine mögliche Interpretation ihrer Elemente deutlich.

Ohne Schwierigkeiten lassen sich auf der Grundlage dieses Modells die Ereignispaare angeben, die parallel zueinander ablaufen können und somit kausal unabhängig sind.

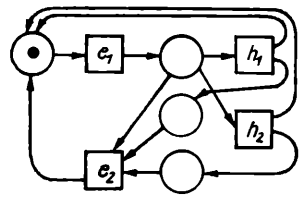
Man muß dazu untersuchen, ob die entsprechende Konstellation der Bedingungen als Fall existiert oder nicht. So ist z.B. $(00'\ 3)$ kein Fall, was der schon a priori bekannten Tatsache der kausalen Abhängigkeit des Ereignisses b_1 vom Ereignis a_1 entspricht. Dagegen gilt $(10'3) \in F$, und somit sind die Ereignisse a_2 und b_1 nebenläufig. Es sei an dieser Stelle darauf hingewiesen, daß Nebenläufigkeit (oder Parallelität) zweier Ereignisse nicht notwendig deren gleichzeitiges Eintreten impliziert, sondern nur diese Möglichkeit einschließt. Postuliert man Ereignisse der Dauer 0 in Verbindung mit der Unteilbarkeit eines Ereignisses, so verschwindet die Gleichzeitigkeit praktisch sogar völlig, weil die Auftrittswahrscheinlichkeit dann ebenfalls 0 ist.

Wegen der weiteren „Nicht-Fälle" $(0\ 1')$, $(2\ 1'\ 3')$ und $(2\ 0'\ 3\ 3')$ sind neben (a_1, b_1) die Ereignispaare $(a_1, b_1,)$ (b_2, a_3) und (b_1, a_3) kausal abhängig, alle anderen aber paarweise nebenläufig. Zeichnet man den gesamten Fallgraphen auf, so entsteht exakt der Automatengraph vom Bild 5.7b. Die Überführbarkeit eines Bedingung–Ereignis-Systems in einen *endlichen Automaten* ist eine wichtige Eigenschaft, vor allem im Hinblick auf die Analyse der Korrektheit einer Beschreibung.

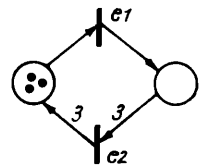
Da die Beschreibungskraft der BE-Systeme zur Beschreibung komplizierter Systeme und Prozesse etwas zu gering ist, werden höhere Netzebenen mit größerer Beschreibungskraft geschaffen. Von diesen sollen im weiteren, auch als Grundlage für die Anwendung zur Steuerung diskreter technischer Prozesse, die sog. Stellen-Transitions-Netze (S/T-Netze, Petri-Netze), angegeben werden. Dabei werden die Bedingungen zu den sog. Stellen verallgemeinert, die jetzt seine bestimmte Anzahl von Marken (Maximum = Kapazität des Platzes) tragen können und somit als Speicher oder Zähler für Marken verwendet werden. Analog dazu erhalten die Verbindungen eine Vielfachheit, die die Anzahl der zu transportierenden Marken angibt. Daraus ergibt sich eine neue Schaltregel für die jetzt als Transitionen bezeichneten Ereignisse des BE-Systems, die u.a. die Markenanzahl berücksichtigt. Außerdem sind jetzt auch Schlingen (Bild 5.9) möglich, die, wie später noch gezeigt wird, zur Realisierung von Testfunktionen sehr wesentlich sind.

Ein anschauliches Beispiel für die Unterschiede in der Beschreibungskraft von BE-Systemen und S/T-Netzen wird im Bild 5.8a und b gezeigt, wobei ein Zähler modulo 3 modelliert wird, d.h., es soll das Verhaltensschema $e_1e_1e_1e_2e_1e_1e_1e_2e_1\ldots$ erzwungen werden. Zur Realisierung als BE-Systeme sind die Hilfsereignisse h_1 und h_2 notwendig, die für den Systemzusammenhang unwesentlich sind. Genauere Ausführungen zu der Netzproblematik findet der interessierte Leser in [5.9] [5.23].

Soll die bisher aufgestellte allgemeine Theorie der Netze auf einem bestimmten Gebiet appliziert werden, so bedarf es einer entsprechend problemangepaßten Interpretation der Netzelemente und einer evtl. notwendigen Modifikation ihres Verhaltens. Damit ent-



e1 h1 e1 h2 e1 e2 ... oder
a) e1 h2 e1 h1 e1 e2 ...



e1 e1 e1 e2 ...
b)

Bild 5.8. Verallgemeinerung der BE-Systeme zu Stellen-Transitionsnetzen
a) BE-System eines Modulo-3-Zählers
b) Darstellung als S/T-Netz

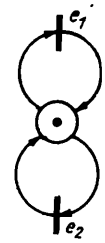


Bild 5.9. Schlingen im S/T-Netz

stehen spezielle Netztheorien, die sich in den letzten Jahren in vielen Gebieten von Naturwissenschaft und Technik geradezu stürmisch entwickelt haben.

Im folgenden sollen zwei Interpretationen vorgestellt werden, die bei der Modellierung und Realisierung von Steuerungen diskreter technischer Prozesse einen großen Teil der

Netzelemente	zugeordnete Steuerungselemente im			
	SN		AKN	
Stellen (Plätze)	z_i y_i k_L	Zustandsvariablen z_i Ausgangsvariablen y_i Koppelsignale k_L		Komponenten des Taskzustandsvektors, externe und interne Nachrichtenpuffer
Transitionen	$f_\mu(\underline{x})$	Zustandsübergänge mit Booleschen Funktionen über den Eingangsvariablen x_i als zusätzliche Schaltbedingungen		Taskzustandsübergänge oder Prozeduren (mit E/A – Fkt. $f(x_i, y_i)$) mit Vergleichsoperationen über globale Variable g_i als zusätzliche Schaltbedingungen Interruptroutinen mit Eingangssignalen x_i als notwendige und hinreichende Schaltbedingungen Zeitfunktionen
		Vergröberung sequentieller Teilautomaten (Netze)		
(Fluß) Verbindungsrelationen		Realisierung der Zustandsübergänge Testfunktionen Zeitfunktionen		Schreibende Funktion auf Taskzustandsvektoren und Nachrichtenpuffer Lesen von Nachrichtenpuffern
Anfangsmarkierung		Initialzustand des Steuerungssystems		

Bild 5.10. Interpretation der S/T-Netz-Elemente für SN und AKN

anfallenden Aufgabenstellungen abdecken können. Die entstehenden Netzklassen werden als sog. *Steuernetze* (SN) bzw. *Ablaufkontrollnetze* (AKN) bezeichnet und dienen der Beschreibung einfacher binärer Steuerungen und deren Hardware- oder Softwarerealisierung (SN) bzw. der Beschreibung von Echtzeitsteuerprogrammen für softwareorientierte Implementierungen allgemeiner Automatisierungsprobleme (AKN).

Bild 5.10 zeigt die Zuordnung der Elemente aus dem jeweils problemspezifischen Beschreibungsraum der beiden Netzinterpretationen zu den Elementen (S, T, V, m_0) des uninterpretierten S/T-Netzes.

Im weiteren folgen einige Erläuterungen zu den einzelnen Netzklassen.

Steuernetze (SN)

Die Darstellung und Beschriftung von Stellen und einfachen Transitionen ist Bild 5.11 zu entnehmen. Bei Makrotransitionen wird mit dem vergebenen Namen ein Teilnetz bezeichnet, das allerdings eine bestimmte Eigenschaft, die sog. Wohlstrukturiertheit [5.24], haben muß. Im wesentlichen bedeutet dies, daß das Teilnetz korrekt (lebendig) ist und

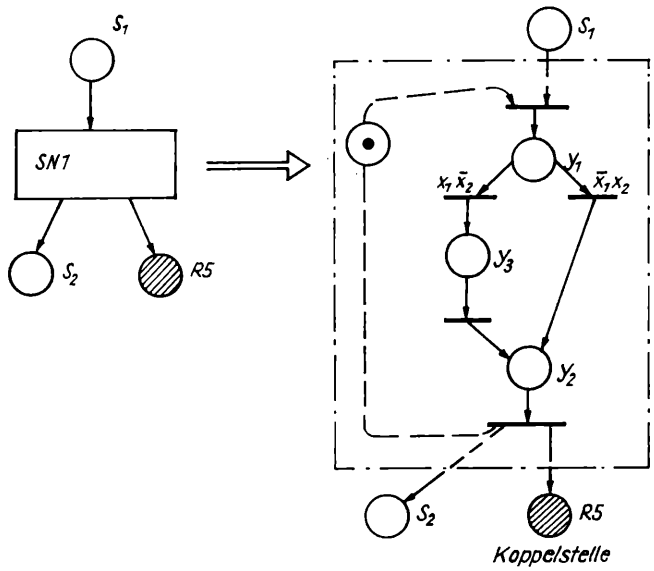


Bild 5.11
Verfeinerung von Netzmakrotransitionen

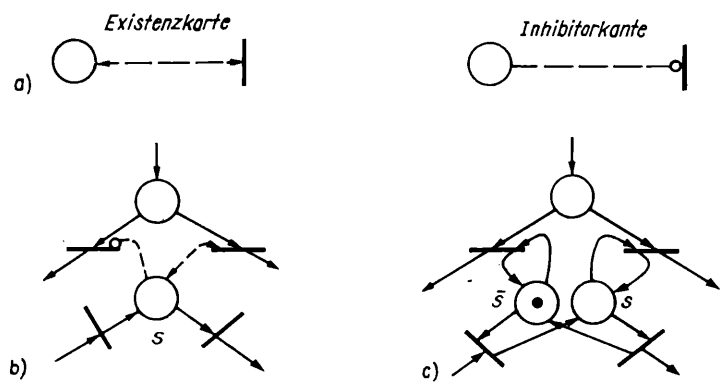


Bild 5.12. Kantenmodifikationen im SN

a) Darstellung der Testkanten; b) Beispiel einer Konfliktlösung mittels Testkanten; c) Rückkonstruktion der Testkanten

genau eine Start- und genau eine Stoptransition besitzt, zwischen denen sich eine Stelle mit der Initialmarkierung befindet, die beim Ersetzen weggelassen wird. Bild 5.11 zeigt eine Beispielkonstruktion. Die Testfunktionen von Verbindungselementen erstrecken sich auf die Überprüfung des Vorhandenseins (Existenzkante) oder Nichtvorhandenseins (Inhibitorkante) einer Marke auf einer beliebigen Stelle. Diese Einschränkung ist möglich, weil in Steuernetzen ohnehin alle Stellen die Kapazität 1 haben, d.h., SN sind (0,1)-markierte Netze. Die Darstellung der genannten Kanten und ihre mögliche Rücküberführung in Elemente des S/T-Netzes wird im Bild 5.12 gezeigt. Dabei wird für die Inhibitorkante eine sog. Co-Stelle (negierte Stelle) eingeführt (Bild 5.12c).

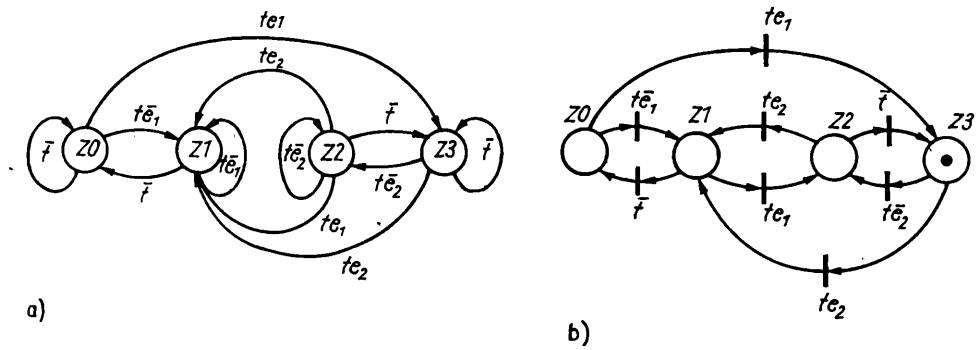


Bild 5.13. Vergleich der Beschreibungsmittel
a) Automatengraph zum PAG von Bild 5.6b; b) Steuernetz zum PAG von Bild 5.6b

Ein kurzer Vergleich der Beschreibungsmittel Steuernetz, Programmablaufgraph und Automatengraph am Beispiel der Supportsteuerung ist dem Bild 5.13 zu entnehmen, wobei der Automatengraph einer speziellen Klasse von Steuernetzen, der Zustandsmaschine, gleichzusetzen ist. Da mit Steuernetzen parallel ablaufende Prozesse in einer geschlossenen Darstellung erfaßbar sind, ergibt sich bei der Nutzung einer Ausgangstabelle zur Darstellung der Stellgliedansteuerung folgende Besonderheit:

In den Spalten sind (wegen der möglichen gleichzeitigen Markierung von Plätzen) nur dann Eintragungen vorzunehmen, wenn ein Platz eine definierte Wirkung auf ein Stell-

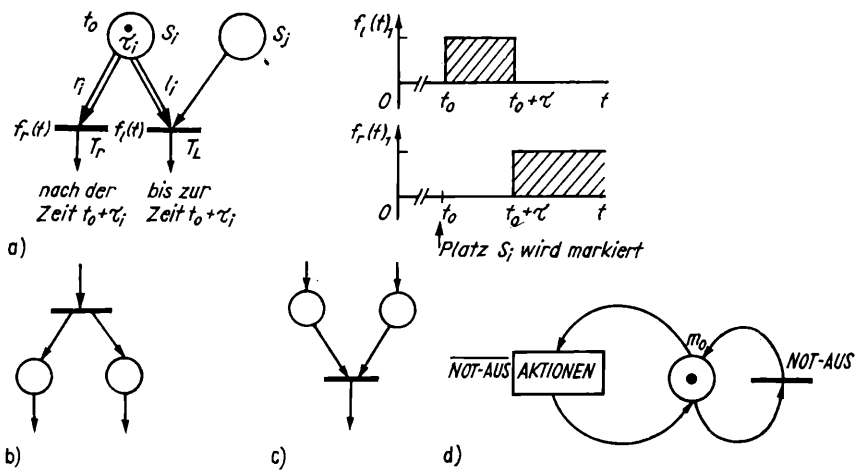


Bild 5.14. Möglichkeiten der Nutzung von Steuernetzen zur Darstellung von a) zeit- und ereignisbedingten Prozessen; b) Eröffnung und c) Synchronisation von Parallelarbeit; d) symbolische Darstellung der Einnahme des Anfangszustands aus jedem beliebigen Zustand heraus

signal haben soll. Dementsprechend sind die Stellsignale nur von markierten Plätzen aus der Ausgangstabelle abzuleiten.

Das Entwurfsmittel Steuernetz bietet auch Möglichkeiten, realisierungsabhängige Besonderheiten einzuarbeiten. Wie im Bild 5.14 die ereignis- und zeitabhängige Schaltfähigkeit von Transitionen erfaßt wird, so können vom jeweiligen Entwerfer auch andere Besonderheiten oder Vereinbarungen eingearbeitet werden, die jedoch den angegebenen Regeln und Eigenschaften entsprechen müssen. Ein Beispiel dafür ist die Einfügung von komplexen logischen Ausdrücken in die Ausgangstabelle, so daß ähnlich dem Programmablaufgraphen ein Mealy-Automat erzeugt wird.

Die Verwirklichung bestimmter Not-Aus-Funktionen technischer Realisierungen bringt bei Einhaltung der Regeln ähnliche Probleme. Hierbei ist es sinnvoll, in einer sehr groben Darstellungsstufe, die dem eigentlichen Steuernetz des Prozeßablaufs überlagert ist, eine generelle Überführung der momentanen Markierung in die Anfangsmarkierung darzustellen (das spart spezielle Knoten und Kanten zum Markenabzug bei Einhaltung der geforderten Netzeigenschaften ein und wird nicht der Netzanalyse unterzogen) bzw. spezielle Wege in Teilnetzen einzuführen (vgl. Beispiel Abschn. 5.4.3. und Bild 5.14d).

Ablaufkontrollnetze (AKN) [5.10]

Im AKN werden sequentielle Teilprozesse (TASK) und deren Verkopplung dargestellt. Die Task bestehen aus internen Stellen (repräsentieren den Tastkzustandsvektor), externen Koppel- (Synchronisations-) Stellen mit der Kapazität 1 und Prozeduren in Form von

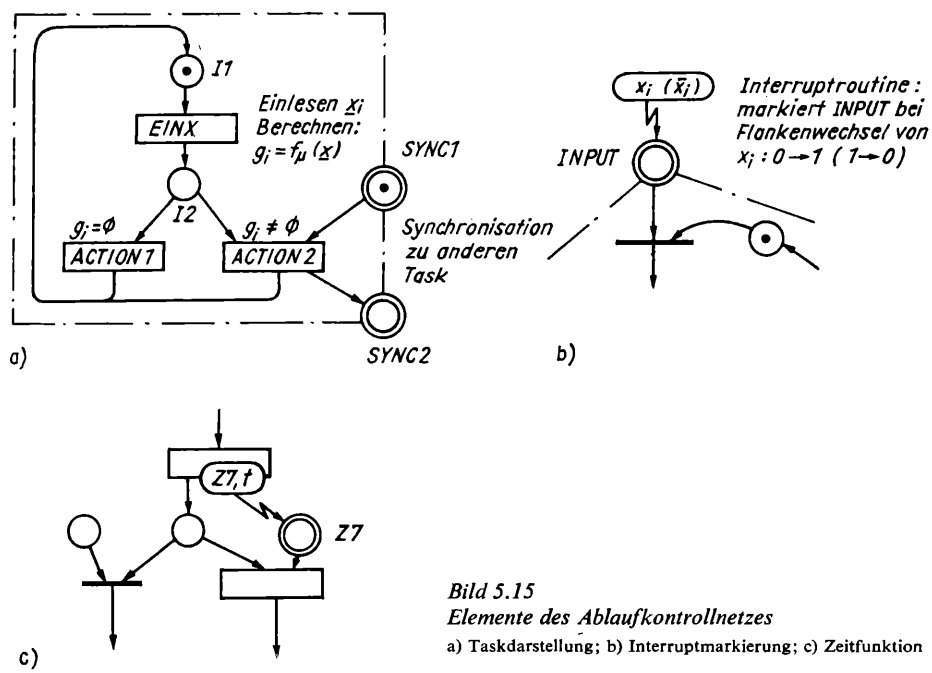


Bild 5.15
Elemente des Ablaufkontrollnetzes
a) Taskdarstellung; b) Interruptmarkierung; c) Zeitfunktion

Aktionstransitionen, die über globale Variable g_i auf den Steuerfluß einwirken können (Bild 5.15a). Außer über die Eingabefunktionen der Prozeduren können Eingangsvariable x_i auch über Interrupt, dargestellt durch Interrupt- (Ereignis-) Transitionen, an die Steuerung gelangen (Bild 5.15b). Bei den Verbindungen gilt prinzipiell das unter den Steuernetzen Gesagte, wobei hier aus implementierungstechnischen Gründen (s. Abschn. 5.5.4.)

auf die Kantenvielfachheit verzichtet wurde. Die Zeitfunktion steckt zum einen implizit in den Laufzeiten der Prozeduren; zum anderen besteht die Möglichkeit, auch innerhalb von Aktionstransitionen speziell spezifizierte Stellen zeitabhängig (verzögert) zu markieren (Bild 5.15c).

5.3. Modellbildung

Beim Entwurf von Steuerungen auf Personalcomputern wird die Überprüfung des Steuerungsentwurfs vielfach nicht am realen Steuerungsobjekt, sondern an einem Modell durchgeführt. Je nach Art der Zielvorstellungen zur Simulation und des Betrachtungsniveaus können verschiedene Simulationssysteme und damit Modellierungsmethoden zur Anwendung kommen.

Für den Bereich der Fertigungssysteme ist der Prozeß im wesentlichen durch Orts- (Lage-) Größen und Energiegrößen gekennzeichnet. Für die Modellierung ist dann zu entscheiden, ob mit der Simulation Aussagen über den Werkstückfluß im Fertigungssystem oder über das Verhalten der ungesteuerten Teile des Fertigungssystems – Roboter, Werkzeugmaschine und Transportsystem – erhalten werden sollen. Während für die Aussagen zum Werkstückfluß nur eine globale Modellierung der Teilsysteme notwendig ist, benötigt man in anderen Fällen detaillierte Aussagen über das Verhalten der Teilsysteme. In diesem Sinne soll unterschieden werden in

- globale Modelle
- lokale Modelle.

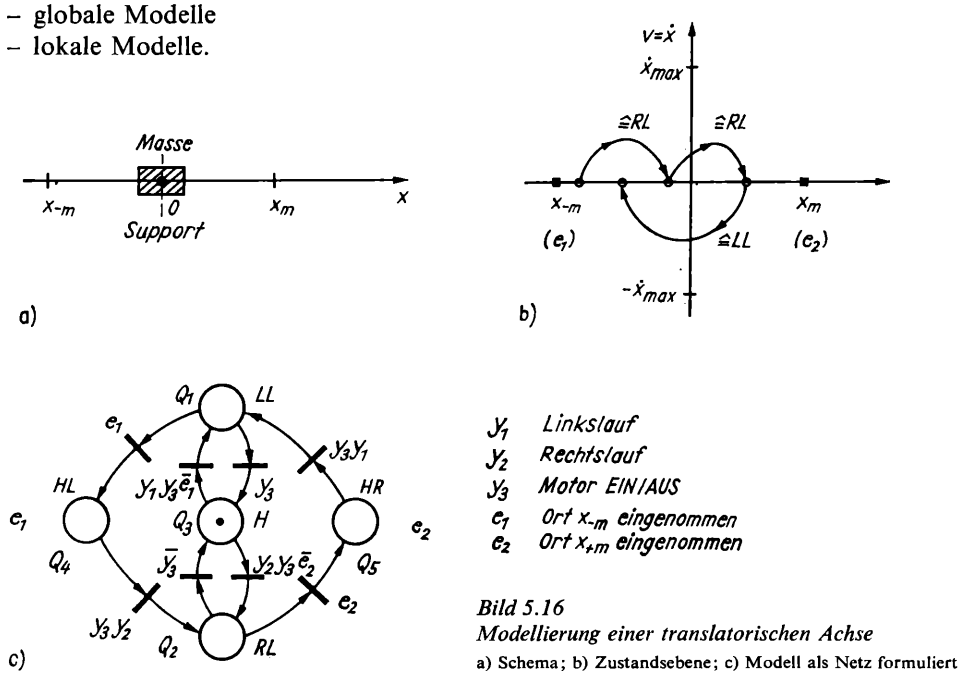


Bild 5.16
Modellierung einer translatorischen Achse
a) Schema; b) Zustandsebene; c) Modell als Netz formuliert

Ein System, das auf der Basis von *globalen Modellen* Simulationsaussagen zu Werkstückfluß und Belegungszeiten bzw. Verweilzeiten liefert, ist das Simulationssystem SIMDIS [5.25]. Da diese Betrachtungen hier nicht im Vordergrund stehen, soll im folgenden nur auf die lokalen Modelle eingegangen werden.

Bei den *lokalen Modellen* wird eine Modellierung der bewegten Funktionseinheiten des zu steuernden Objekts angestrebt. Im Fall der Roboter und Werkzeugmaschinen, allgemein bei Fertigungssystemen unterscheidet man [5.2]:

- lagebestimmte Funktionsseinheiten
  Lagebestimmte Funktionseinheiten sind vor allem Vorschubachsen – translatorische Achsen, rotatorische Achsen usw. – und Transporteinrichtungen. Sie besitzen ohne Energiezufuhr beliebig viele stabile Lagen. Die charakterisierende Größe ist immer ein Orts–Zeit-Verlauf.
- energetisch bestimmte Funktionseinheiten.
  Energetisch bestimmte Funktionseinheiten sind vor allem elektrische Antriebssysteme und hydraulische Systeme. Die charakterisierende Größe ist stets eine Energie – Energie des Magnetfelds usw. – oder eine Ersatzgröße, die die Energie ausreichend bestimmt.

Die Modellbildung soll an einem einfachen Beispiel einer translatorischen Achse mit zwei Endlagenschaltern erläutert werden. Im Abschn. 4. wurde dafür die Bewegungsgleichung diskutiert und die Zustandstrajektorien $f(\dot{x}, x)$ für eine optimale Steuerung im Bild 5.25 dargestellt. Bei einer gröberen Betrachtung der Funktionseinheit (Bild 5.16a) interessieren nur ausgewählte Gebiete bzw. Punkte in der Zustandsebene. Im Beispiel sind dies der rechte Randpunkt x_m, der linke Randpunkt x_{-m} und die x-Achse für den Zustand „Ruhe". Die Quadranten I und II stehen für den Zustand „Linksbewegung" und die Quadranten III und IV für den Zustand „Rechtsbewegung" (Bild 5.16b). Damit gewinnt man bei einer Beschreibung mit einem steuerungstechnisch interpretierten Netz das Bild 5.16b. Eine Steuerung für diese Achse kann die im Bild 5.13 angegebene sein.

5.4. Funktioneller Entwurf

5.4.1. Überblick

Der Stand des funktionellen Entwurfs ist gegenwärtig dadurch gekennzeichnet, daß auf ein Modell des Steuerungsobjekts verzichtet wird und der funktionelle Entwurf aufgrund der Formulierung der Steuerungsaufgabe (s. Bild 5.3) erstellt wird.

Wenn das Verhalten des Steuerungsobjekts von Fall zu Fall doch beschrieben wird, so wird in der Regel die Beschreibung von Teilabläufen der erst noch zu entwerfenden Steuerung mit der Beschreibung des Steuerungsobjekts stark miteinander vermischt. Im folgenden soll der funktionelle Entwurf ohne Erstellung eines Modells behandelt werden.

Es sei nochmals erwähnt, daß von den technologischen Anforderungen ausgehende genaue Kenntnisse der gewünschten Bewegungsabläufe, Betriebsarten, Funktionen und Verriegelungen vorliegen müssen. Daran schließen sich Vorstellungen zur hierarchischen Strukturierung an. Diese Strukturierung ist jedoch in Wechselwirkung mit der Realisierungsphase zu stellen.

5.4.2. Entwurf mit Steuernetzen oder PAG

Der Umfang, die angestrebte Servicefreundlichkeit und die geforderte Änderbarkeit von Robotersteuerungen und Maschinensteuerungen bringen es mit sich, daß eine Zerlegung in Teilsteuerungen unumgänglich ist. Der Entwurf der Steuerungshierarchie und der Festlegung des funktionellen Inhalts der Teilsteuerung erfolgt intuitiv. Bild 5.17 zeigt eine

prinzipielle bewährte Form für Ablaufsteuerungen. Gesichtspunkte für eine Zerlegung in Funktionsgruppen bzw. Moduln ergeben sich aus

- konstruktiv-technologischer Sicht
  - Ableitung von Elementarmoduln (Betätigung der Greifer, Verfahren einer Achse, Positionieren auf einen numerisch vorgegebenen Sollwert usw.;
  - Realisierung technologischer Teilfunktionen unter Einbeziehung von Elementarmoduln (Entnahme eines Rohteils aus der Palette, Einlegen eines Rohteils in die Maschine usw.);
  - Zusammensetzung eines gesamten technologischen Funktions- bzw. Bewegungszyklus aus Teilfunktionen mit Hilfe eines Koordinators;
  - Auswahl verschiedener technologischer Funktions- bzw. Bewegungszyklen durch Einführung von Subkoordinatoren, die den Koordinator modifizieren, indem sie verschiedene Wege im Koordinator auswählen (Bestimmung des aktuellen Palettenplatzes, Auswahl der Maschinenposition, Auswahl bestimmter Bewegungskombinationen usw.),
- steuerungstechnischer Sicht
  - Funktionsgruppe Steuerung EIN (Herstellen der Betriebsbereitschaft);
  - Bildung steuerungstechnischer Signale (Startbedingung, Grundstellung, Vermeidung von Kollisionen, Havarien);
  - Reihenfolge des Zuschaltens zentraler Einrichtungen (Hydraulik, Druckluft, elektrischer Leistungsteil);
  - Realisierung von Not-Aus;
  - Betriebsartenauswahl (Start–Stop–Halt, 1 × -Durchlauf, Anfangs- und Endebehandhandlung, Automatik, Tipp-, Handbetrieb);
  - Fehleranzeigen;
  - Reaktionen auf Fehler;
  - Beherrschung zeitlicher Effekte (Impulsbildung, Direkteingaben, Interruptverarbeitung, zeitliche Überdeckung von Teilmoduln);
  - Bestimmung der Reihenfolge der Abarbeitung der Programmteiler (entfällt bei Polingbetrieb),
- dialogtechnischer Sicht
  Die zugehörige Dialogsteuerung soll sich dabei nicht auf die normale Roboterbedienung bei ausgetesteten Programmen beziehen, sondern Funktionen enthalten, die die Erstinbetriebnahme, Fehlersuche und -lokalisierung, Programmkorrektur und -änderung, Programmeingabe, Programmdokumentation beinhalten. Stichwörter sind hier
  - Diagnoseprogramme,
  - nachladbares oder residentes Dialogprogramm,
  - Programmiergerät,
  - Teach-in-Programmierung.

Für die Hierarchiebildung und Zergliederung der Gesamtfunktion gibt es beim PAG und beim Steuernetz keine Unterschiede. Auch werden in beiden Fällen Zustandsmaschinen zur Beschreibung elementarer Abläufe benutzt. Unterschiede bestehen bei der Kopplung der elementaren PAGen und der Steuernetze:

PAG: Kopplung zwischen PAG wird durch Setzen von Koppelvariablen über die Operationen und Abfragen dieser in den anderen PAG in den Verzweigungen
SN: Kopplung durch Überlagerung der Zustandsmaschinen.

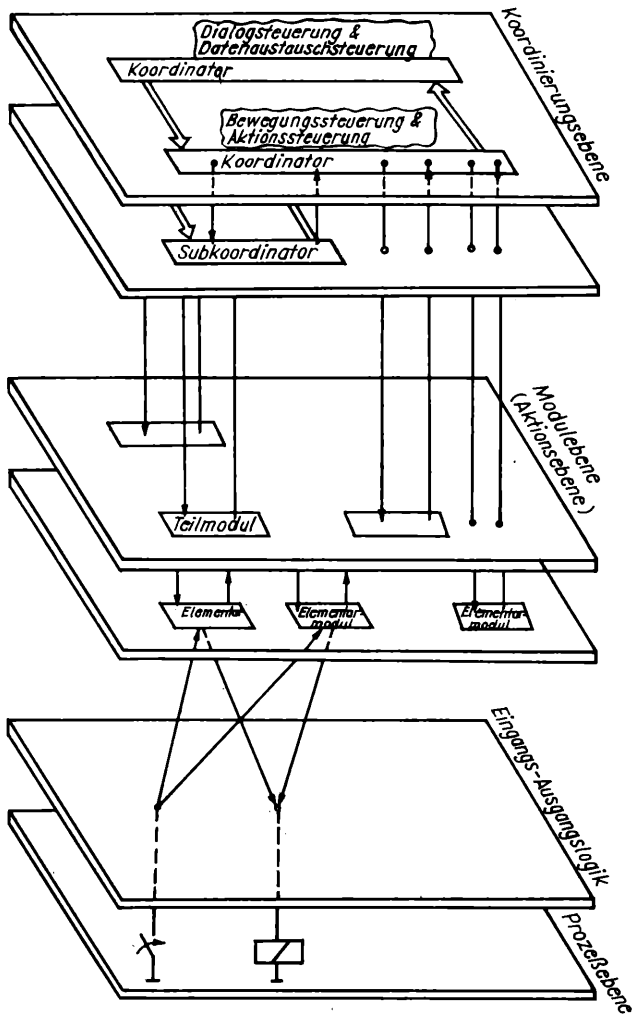


Bild 5.17
Entwurfshierarchie für Ablaufsteuerungen

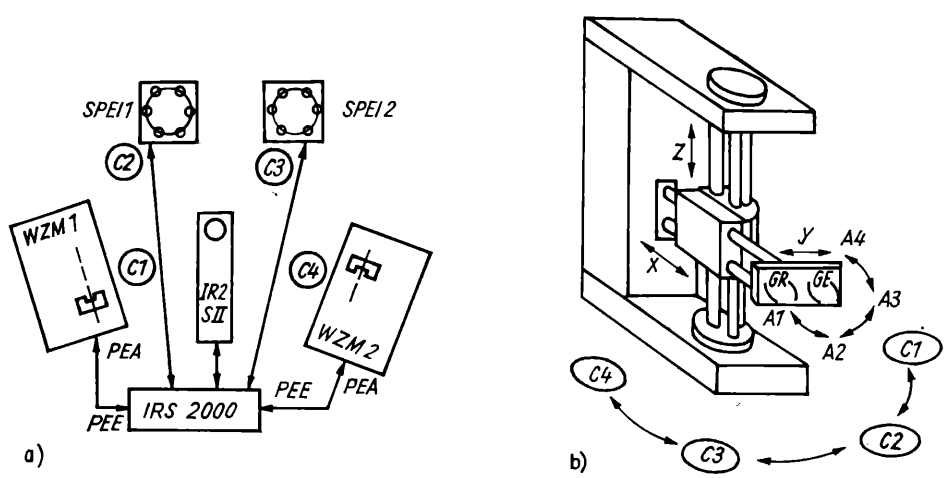


Bild 5.18. Technologische Einheit

a) Anordnung der Komponenten in der technologischen Einheit; b) prinzipieller Aufbau des IR2/SII

Im folgenden soll an einem Demonstrationsbeispiel die Anwendung des Entwurfsverfahrens mit Steuernetzen erläutert werden [5.14]. Dieses Beispiel entstand auf der Grundlage von technisch realisierten Programmen für den Beschickungsroboter IR 2/S II [5.15] [5.16]. Dieser Roboter ist in eine technologische Einheit (TE) mit zwei Werkzeugmaschinen (WZM 1, 2) und zwei Werkstückspeichern (SPEI 1, 2) integriert und dient der Beschickung der WZM mit Rohteilen (Bild 5.18).

Die Steuerung des Roboters übernimmt eine speicherprogrammierbare Steuerung (Typ IRS 2000), die Boolesche Gleichungen verarbeitet und somit eine Punkt-zu-Punkt-Steuerung ermöglicht. Positions- und Geschwindigkeitsregelungen sind Hardwarefunktionen. Dabei ist die Anzahl der möglichen Positionen je Achse:

- C-Achse 4
- A-Achse 4
- X-Achse 10 (Sollwerte K1X bis K10X, K10X = X-Achse hinten)
- Z-Achse 20 (Sollwerte K1Z bis K20Z, K20Z = Z-Achse oben, K19Z = Aushub ΔZ)
- Y-Achse 2.

An der *A*-Achse sind zwei Greifer für die Rohteil- (GR) und Fertigteilhandhabung (GF) installiert. Die Werkstückspeicher besitzen sechs Ebenen mit jeweils sechs Werkstückplätzen und sind vor Entnahme eines Teiles jeweils um eine Position weiterzudrehen (takten). Aufgabe des Roboters ist es, die anfordernde Werkzeugmaschine (WZM 1 oder 2) mit Rohteilen vom entsprechenden Werkstückspeicher (SPEI 1 oder 2) zu beschicken (WZM leitet Fertigteil selbständig weiter).

Der Abbruch der Beschickung erfolgt nach Erkennung des letzten Werkstücks des Loses (LWL), woraufhin der Roboter seine definierte Grundstellung einzunehmen hat.

Als mögliche Betriebsarten wurden der Automatikbetrieb AUT (Koordinator organisiert die Bewegungsfolge) und der Handbetrieb HA (Erfassung in den entsprechenden Elementarmoduln) realisiert, wobei der Automatikbetrieb durch die Anwahl von STOP oder HALT (am Zyklusende) unterbrochen werden kann.

Die Steuerung ist aus insgesamt 29 Moduln (Funktionsgruppen FG) konzipierbar und kann folgendermaßen hierarchisch gegliedert werden:

Steuerungstechnische Funktionsgruppen

FG 1	Steuerung EIN (zur Auswertung von Not-Aus-Signalen)
FG 2	Hilfsenergie zuschalten (Hydraulik EIN)
FG 3	Drucküberwachung Hydraulik Pumpe 2
FG 4	Schutzgitter – Kontrolle und Betätigung
FG 5 bis 8	Kontrollen, Fehlermeldungen
FG 9	Kontrolle der Grundstellung des Roboters (GS)
FG 10	Betriebsartenauswahl und -realisierung,

Konstruktiv-technologische Funktionsgruppen

- Koordinator = FG STW (Steuerwerk)
- Subkoordinatoren
  - FG 21 Unterscheidung der Maschinen (WZM)
  - FG 22 Auswahl der Speicher
  - FG 23 letztes Werkstück des Speichers/Loses (LWS/LWL),
- Teilmoduln
  - FG 1/1 Entnahme Rohteil aus SPEI
  - FG 43 Rohteil in WZM einlegen

- Elementarmoduln
  Bewegungsmoduln
  FG 11 Schwenken C-Achse
  FG 12 Schwenken A-Achse
  FG 13 Y-Achse
  FG 15/1,2 Greifer Roh-/Fertigteil
  FG 16 X-Achse Handbetrieb
  FG 17 Z-Achse Handbetrieb
  FG 18 Aushub ΔZ Handbetrieb
  Zähler
  FG 19/1,2 Ebenenzähler der SPEI 1, 2
  FG 20/1,2 Rohteilzähler für WZM 1, 2
  Peripheriemodul
  FG 14/1,2 Takten der SPEI 1, 2.

Der Kopplung Robotersteuerung–WZM dienen die Signale PEA (IRS → WZM, aktiviert Bewegungen in WZM) und PEE (WZM → IRS, meldet abgeschlossene Aktivitäten),

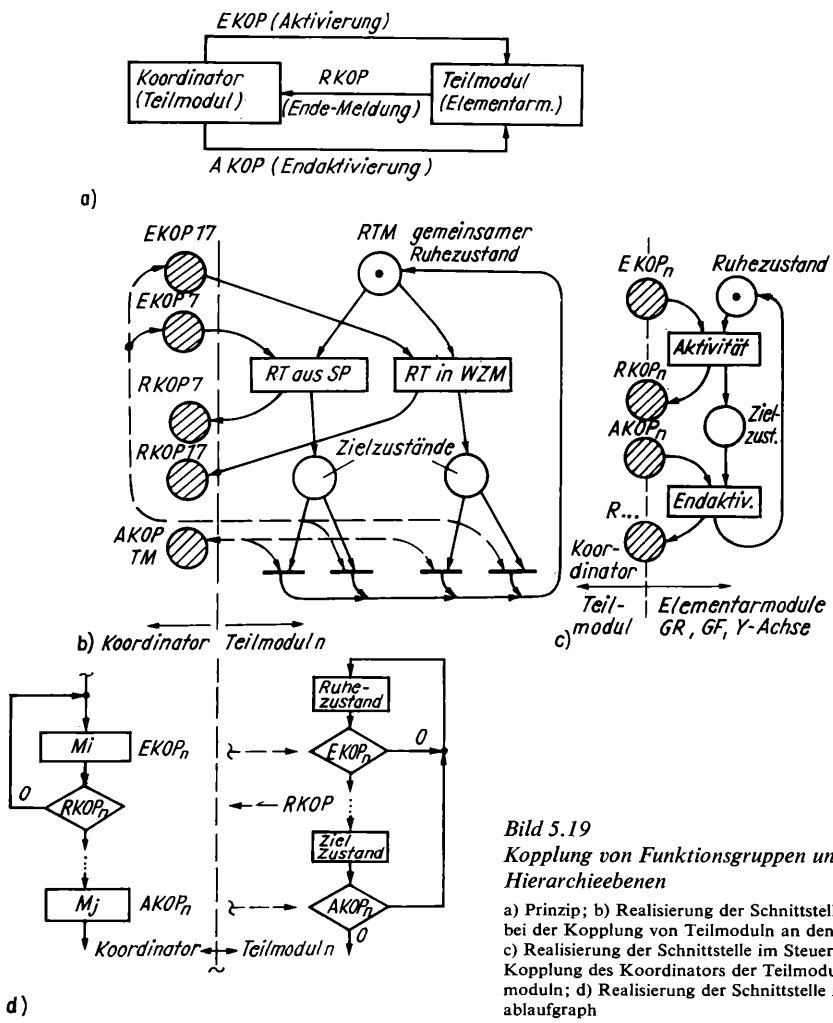


Bild 5.19
Kopplung von Funktionsgruppen unterschiedlicher Hierarchieebenen

a) Prinzip; b) Realisierung der Schnittstelle im Steuernetz bei der Kopplung von Teilmoduln an den Koordinator; c) Realisierung der Schnittstelle im Steuernetz bei der Kopplung des Koordinators der Teilmoduln mit Elementarmoduln; d) Realisierung der Schnittstelle im Programmablaufgraph

die in Moduln verschiedener Hierarchieebenen gebildet bzw. getestet werden (s. auch Bild 5.18a). Eine Dialogsteuerung existiert in diesem Anwendungsfall nicht.

Um die Funktionsgruppen unterschiedlicher Hierarchieebenen verbinden zu können, werden in diesem Beispiel Koppelsignale (Plätze) eingeführt (Bild 5.19). Diese Signale sind

EKOP Einkoppeln von Teil- oder Elementarmoduln (Aktivierung)
AKOP Auskopplung von Teil- oder Elementarmoduln (Entaktivierung)
RKOP Rückmeldung des aktivierten Moduls an den aufrufenden Modul.

Diese Koppelbeziehungen können in den verschiedenen Entwurfsmitteln unterschiedlich realisiert werden. Daraus ergibt sich die Möglichkeit, in den verschiedenen Hierarchieebenen gleiche oder unterschiedliche Entwurfsmittel anzuwenden. Beispielsweise können Koppelsignale im PAG als Ausgangssignale von Zuständen des aufrufenden PAG gebildet und in Übergangsbedingungen des aktivierten PAG abgefragt werden.

Dagegen sind im Steuernetz die Koppelsignale als Plätze (Koppelplätze) dargestellt und somit mit einem Markentransport verbunden. Zu erwähnen ist noch, daß in dieser Art die Koppelsignale nur in der Betriebsart Automatik, also bei Aktivitätssteuerung durch den Koordinator, wirksam sind.

Die Betriebsart Hand wird in den entpsrechenden Elementarmoduln durch besondere Wege im Graphen realisiert. Daraus ergibt sich auch die Notwendigkeit, daß die Elementarmoduln in sich abgeschlossene Funktionsteile bilden (bei Steuernetzentwurf zu beachten).

Im folgenden wird aus den unterschiedlichen Hierarchieebenen je eine Funktionsgruppe vorgestellt, wobei in diesen Entwürfen die Plätze und Transitionen nicht durchnumeriert, sondern mit mnemotechnischen Bezeichnungen versehen sind. Für Transitionen ohne angegebene Schaltbedingung wird für diese der logische Wert 1 vereinbart.

Steuerungstechnische Funktionsgruppen

FG 1 Not-Aus

Diese Funktionsgruppe (FG) erfaßt alle zu einem Not-Aus führenden Signale (RES 7) und erlaubt in Abhängigkeit davon das Einschalten der Steuerung. Die eingeschaltete Steuerung ist Voraussetzung für die Funktion *aller* weiteren Funktionsteile (Bild 5.20).

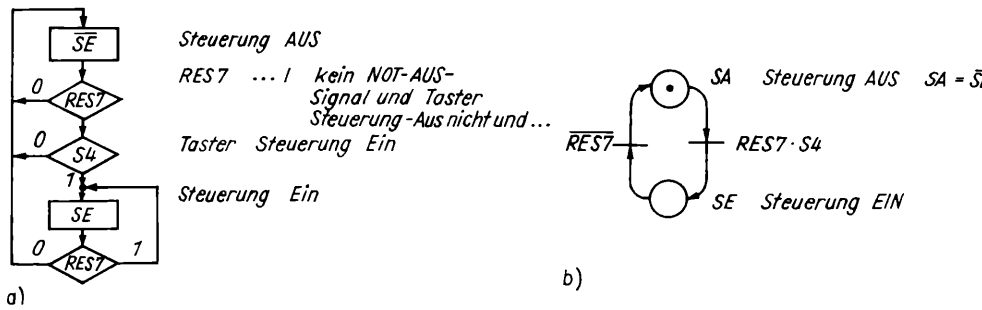


Bild 5.20. Erfassung von Not-Aus-Situationen
a) Programmablaufgraph; b) Steuernetz

Konstruktiv-technologische Funktionsgruppen

- Koordinator Steuerwerk (STW)

Das Steuerwerk koordiniert die Reihenfolge der zur WZM-Beschickung notwendigen Bewegungen und wählt in Auswertung der Markierung von Subkoordinatorplätzen den

geforderten Zyklus aus (Bild 5.21). Es ist als eine Folge von Zuständen (Plätzen M1 bis M10) realisiert, die Elementar- und Teilmoduln aktivieren und nach erfolgter Aktivität (Rückkoppelsignal) weiterschalten. Die Plätze M1 bis M10 sind einerseits notwendig zur Unterscheidung gleicher Aktivitäten (z. B. Schwenken auf C2) bei Nutzung der gleichen

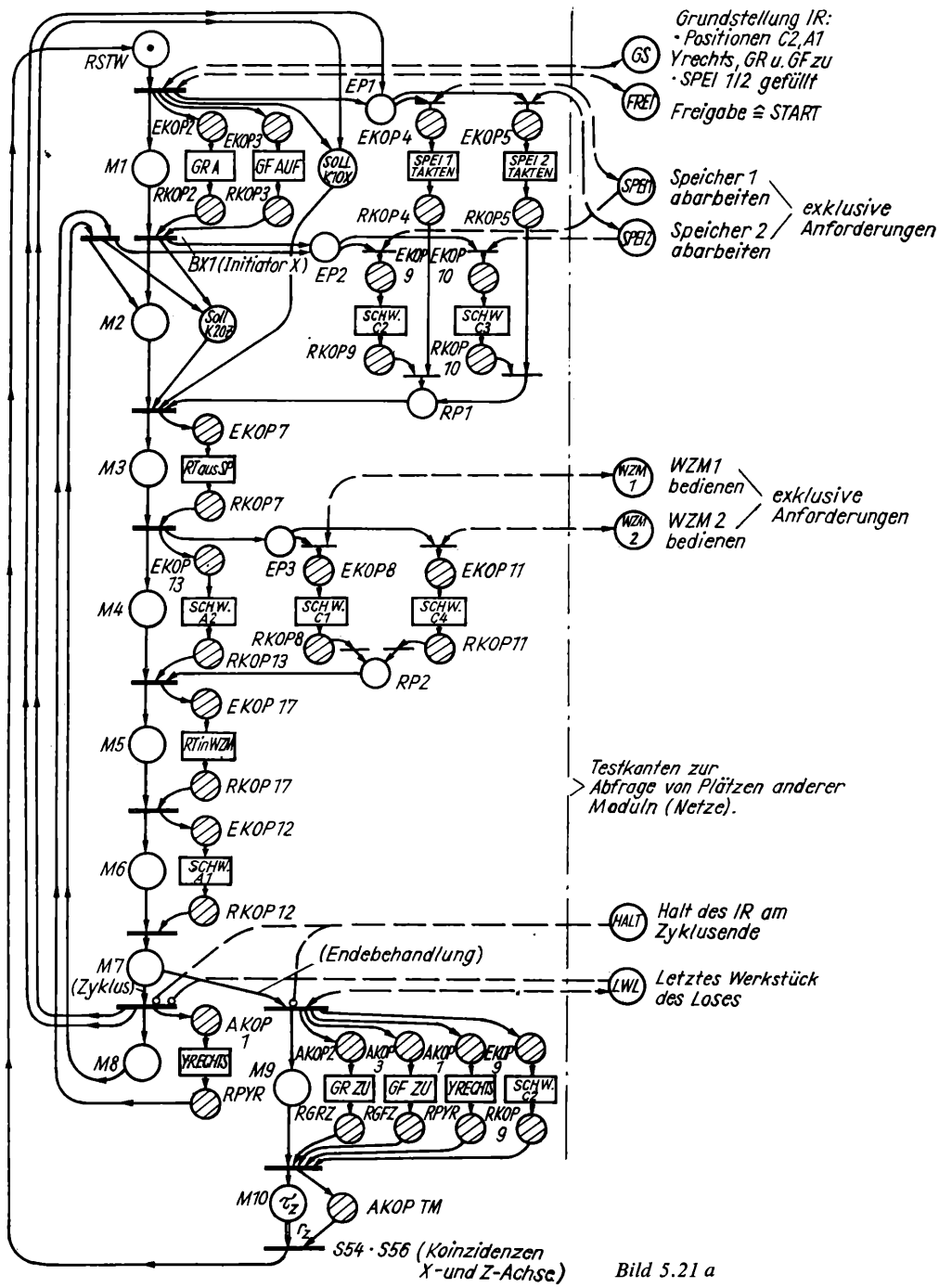
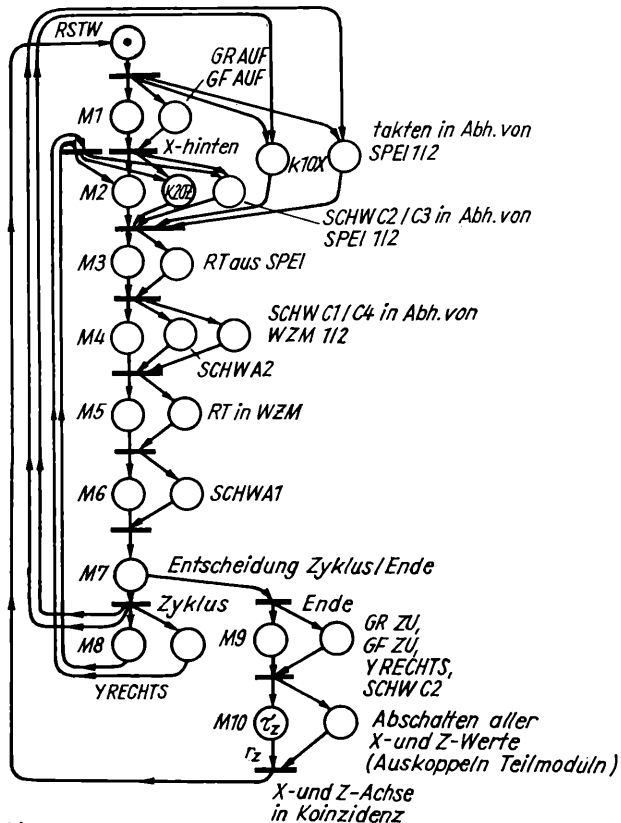


Bild 5.21 a



b)

Bild 5.21
Struktur des Koordinators
a) detailliertes Steuernetz
b) vergröbertes Steuernetz

Koppelplätze und andererseits günstig zur Erkennung des Fortschritts in der Abarbeitung der Beschickungstechnologie.

In einer Grobdarstellung wurden unter Anwendung von Umformungsregeln die Aktionstransitionen und deren Koppelplätze zu Plätzen mit der Bezeichnung der jeweiligen Aktion zusammengefaßt. Damit konnte eine übersichtliche, aber wenig detaillierte Darstellung geschaffen werden. Zu erwähnen ist, daß bei einem Not-Aus-Signal (Steuerung AUS) die Anfangsmarkierung des Koordinators eingenommen wird.

- Subkoordinatoren

Die Funktionsgruppen dieser Ebene dienen der Auswahl verschiedener Wege im Steuerwerk bzw. der Auswahl von Koppelsignalen, die das Steuerwerk aussendet bzw. empfängt. Da diese Funktionsgruppen aus speziellen technologischen Forderungen des Steuerungsanwenders resultieren und mittels einfacher PAG bzw. Steuernetze ohne Parallelität erzeugt werden können, wird an dieser Stelle auf eine detaillierte Darstellung verzichtet.

- Teilmoduln

Bei der Realisierung der TM (technologische Teilmoduln) mittels Netze können die Einkoppelsignale folgender Module zum Entaktivieren des jeweiligen genutzt werden. Zum Abschluß der Werkstückhandhabung (Endezustand des Koordinators) wird dann ein für alle TM gemeinsames Auskoppelsignal (AKOP TM) gebildet, das den jeweiligen Modul entaktiviert. Dieses Koppelprinzip ist im Bild 5.19b dargestellt.

Auf der Hierarchieebene der Teilmoduln organisiert die FG 1/1 Entnahme Rohteil aus Speicher folgenden Bewegungsablauf:

1. Z-Position der Speicherebene (zusammen mit FG 19/1,2 Ebenenzähler),
2. X-Position des Speichers (zusammen mit FG 22 Speicherauswahl),
3. Greifer schließen,
4. Aushub ΔZ (Wert K19Z),
5. X-Achse in hintere Endlage verfahren (Wert K10X),
6. Z-Achse in obere Endlage verfahren (Wert K20Z), dabei Aushub aus (Bild 5.19a).

Die Funktion kann durch vier parallele Zweige eines Netzes realisiert werden (Bild 5.22b), wobei von den Plätzen aus die entsprechenden z- und x-Werte aufgerufen werden. In diese FG ist der Betriebszustand STOP (keine weitere Bewegung bis zur Aufhebung von STOP) eingearbeitet. Die Plätze R1A1, R1A2, R1A3 bedeuten den sog. Ruhezustand der einzelnen Aktionen.

- Elementarmoduln

Zur Erläuterung von Elementarmoduln wird auf den Bewegungsmodul FG 15/1 Rohteilgreifer (GR) eingegangen (Bild 5.23). Diese FG hat die Betriebsarten Automatik mit den Koppelbedingungen

- EKOP2 Greifer öffnen
- RKOP2 Greifer ist auf
- AKOP2 Greifer schließen
- RGRZ Greifer ist geschlossen

und den Handbetrieb bei Abfrage der Taster

- S25 Greifer öffnen
- S61 Greifer schließen

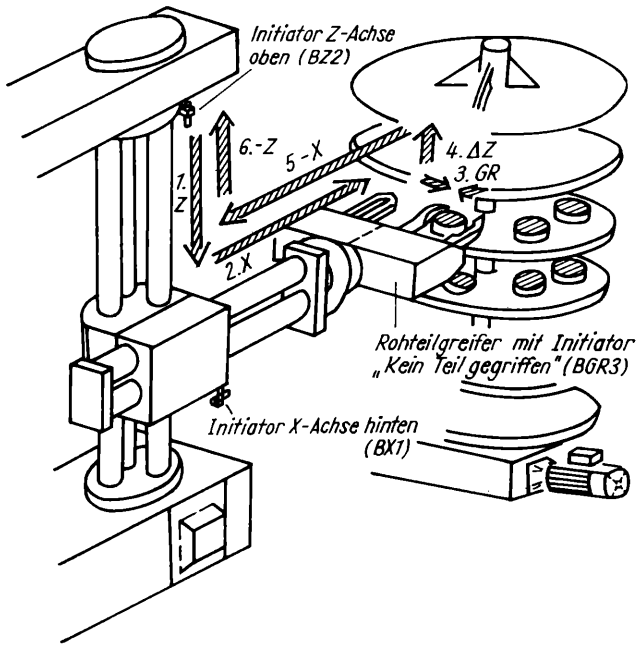
zu enthalten.

Als Grundstellung ist der geschlossene Greifer zu realisieren. Zur Kontrolle von erfolgten Greiferbewegungen wird neben dem Druckmelder S51 ein Zeitbaustein (τ_G) ausgewertet. Diese FG hat auch zu überwachen, ob ein Teil gegriffen wurde (Melder BGR3). Während dazu bei der PAG-Methode ein zusätzlicher PAG (KEINT) geschaffen werden müßte, ist diese Kontrolle im Steuernetz mit enthalten. Im Normalfall wird der Greifer durch Betätigung des Magnets YGR geöffnet und geht aus dem Zustand GRZ (GR zu) über DGR (Druck auf GR) in GRA (GR auf) über. Zum Schließen wird der Magnet YGR geschlossen; das Netz geht in GRS (GR schließen) über. Wird im Automatikbetrieb bei Markierung dieses Platzes der Melder BGR3 betätigt, so werden die Plätze KEIN T (kein Teil gegriffen) und DGR eingenommen, d.h., der Greifer wird wieder geöffnet. Aus der Markierung GRA und KEINT kann man den Greifer nur wieder schließen durch Betätigen der Starttaste (S6) im Automatikbetrieb oder durch Übergang zum Handbetrieb.

In der dem funktionellen Entwurf folgenden Phase des strukturellen Entwurfs muß noch die Ausgangslogik geschaffen werden. Diese stellt die Beziehungen der Steuerungszustände zu den Ausgangssignalen her. Dabei ist zu beachten, daß die Ausgangssignale aus

- Zuständen (Plätzen) eines PAG (Netzes),
- Zuständen (Plätzen) mehrerer PAG (Netze),
- Zuständen (Plätzen) und Eingangssignalen

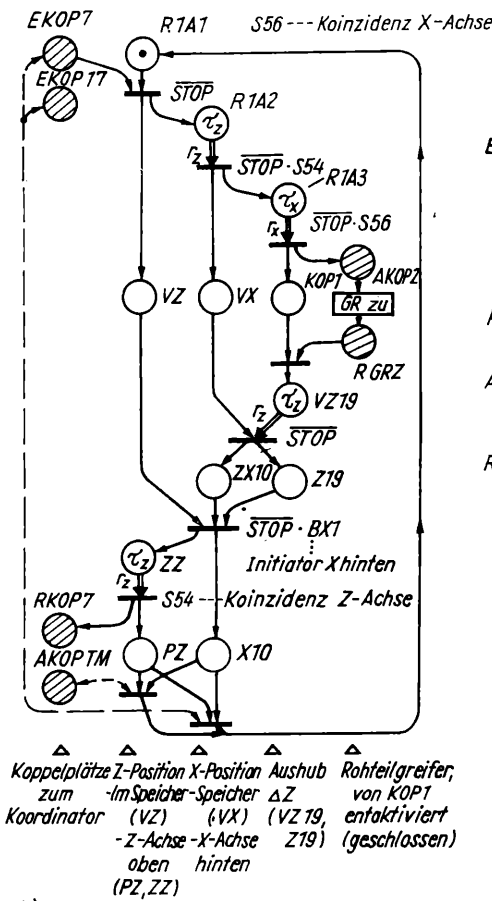
resultieren können.



a)

Bild 5.22
Entwurf des Teilmoduls FG 1/1: Entnahme des Rohteils aus dem Speicher

a) Skizze des Bewegungsablaufs
b) Steuernetz für den Bewegungsablauf



b)

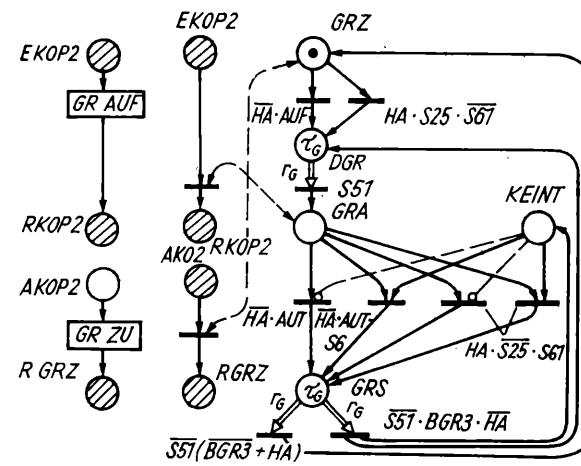


Bild 5.23
Entwurf des Elementarmoduls FG 15/1: Rohteilgreifer

a) Prinzip; b) detailliertes Steuernetz
Ausgabefunktion des Magnetventils: YGF = DGR ∨ GRA

5.4.3. Entwurf mit Ablaufkontrollnetzen

Das Beschreibungsmittel AKN impliziert eine taskorientierte Herangehensweise, d.h., eine Gesamtaufgabe muß zunächst funktionell in Teilaufgaben zerlegt werden, die zwar untereinander parallel, selbst aber nur rein sequentiell gestaltet sein dürfen. Diese Teilfunktionen (Rechenprozesse, Tasks) erhält man zweckmäßigerweise durch Dekomposition, wobei mit Hierarchie und Datenrelationen gearbeitet werden sollte. Sind die einzelnen Task funktionell umrissen, können sie einzeln in zwei Ebenen entworfen werden,

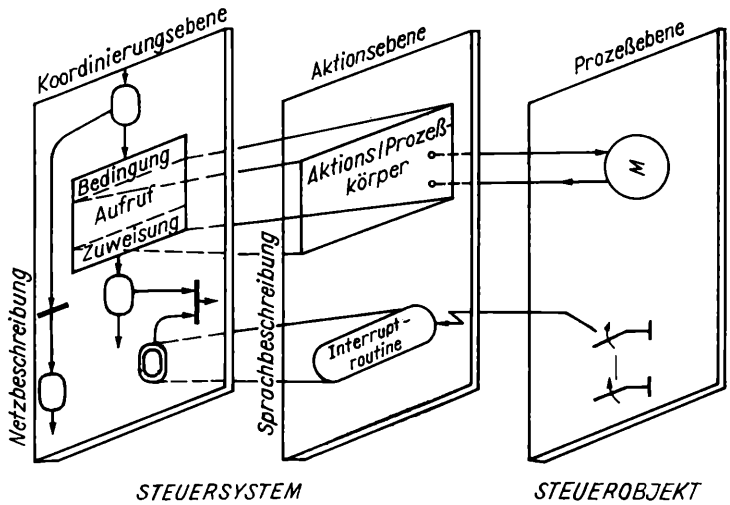


Bild 5.24
Entwurfshierarchie bei Anwendung von AkN

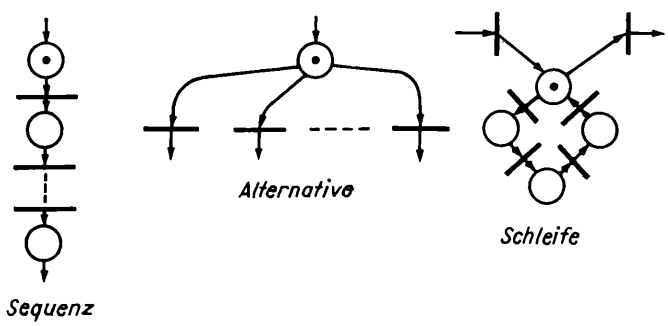


Bild 5.25
Darstellung sequentieller Ablaufkontrollstrukturen

d.h. in der Ebene der Koordinierungssteuerung als AKN und in der Ebene der Aktionssteuerung als sequentielle Taskstücke (Aktionen) und Interruptroutinen (Bild 5.24). Der funktionelle Inhalt der Aktionen wird aus dem Datenfluß heraus fixiert, indem man synchronisationsfreie Abschnitte eines Task (d.h. ohne steuer- oder datenflußmäßige Verkopplung zu anderen Task) zu separaten (Aktions-)Prozeduren deklariert. Die sequentielle Aneinanderreihung dieser Prozeduren, d.h. unter Verwendung der sequentiellen Ablaufkontrollstrukturen Sequenz, Alternative und Schleife (Bild 5.25), ergibt wieder den ursprünglichen Task. Zwischen den einzelnen Prozeduren – und nur dort – findet die Intertaskkommunikation statt, die somit vollständig im AKN beschreibbar ist. Neben diesem Vorteil der eindeutigen Trennung von Steuer- und Operationsteil und der damit gegebenen Möglichkeit des getrennten Entwurfs mit völlig anderen (angepaßten) Beschreibungsmitteln bietet diese Methodologie auch eine gute Basis für die Aufgaben-

verteilung in einem Bearbeitungskollektiv. Dabei sind bei der Konzeption der Prozeduren und Interruptroutinen neben fest definierten Ein- und Austrittsstellen folgende Schnittstellen bezüglich der Koordinierungssteuerung festzulegen:

- Unterbrechbarkeit
  ACTION unterbrechbare Prozedur
  PROCESS nichtunterbrechbare Prozedur
  für ACTION
  - maximale Stapeltiefe (für die Definition eines eigenen Stack)
  - Registerrettungsmodus.
- Parameterübergabe
  Wird in der Prozedur eine für die Koordinierungssteuerung relevante Information erzeugt, so wird diese als globale Variable (s. Zuweisung im Bild 5.25) behandelt, da ja im AKN Ausdrücke über globale Variable als zusätzliche Schaltbedingung an die Transitionen geschrieben werden können, oder es besteht die Möglichkeit der Markierung bestimmter Stellen (Typ TIMED, INPUT), was z.B. der Erfassung externer Ereignisse über IR dient.

Das Beschreibungsmittel für die Prozeduren und Interruptroutinen ist aus der Menge der Mittel für die sequentielle Programierung frei auswählbar und soll hier nicht Gegenstand weiterer Betrachtungen sein.

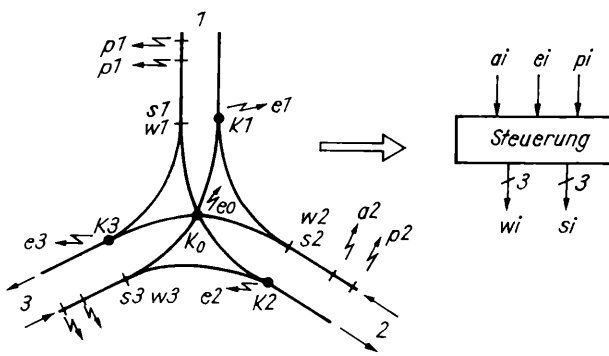


Bild 5.26
Transportbeispiel
Ki Kreuzungspunkte; *Wi* Weichen; *Si* Stopstellen Signale: ai TE-Annäherung aus Richtung *i*; ei TE-Entfernung in Richtung *i*; pi Richtungspuffer *i* voll, wi Weichenstellsignale (Richtung *i*); si Freigabesignale (Richtung *i*)

Das gesamte Entwurfsvorgehen mit den Ablaufkontrollnetzen soll im folgenden an einem komplexeren Beispiel verdeutlicht werden. Bild 5.26 zeigt einen Ausschnitt einer Transportsteuerung, bei der Transporteinheiten (Lastgehänge) kollisionsfrei über die abgebildete 3-Weg-Kreuzung zu leiten sind. Jede Transporteinheit (TE) besitzt eine Kennung, aus der die gewünschte Zielrichtung durch die Steuerung ermittelbar ist, sobald sich die TE innerhalb der den Stopstellen S_i vorgelagerten Pufferstrecken befindet. Befährt eine TE einen solchen Abschnitt, wird die Steuerung mit dem Annäherungssignal a_i beaufschlagt. Das Verlassen der Kreuzung wird über Entfernungssignale e_i gemeldet, die dann aktiv werden, wenn die Kreuzungspunkte K_i (i = 1 bis 3) von der TE passiert wurden.

Zusätzliche Forderungen an die Steuerung sind:

a) Realisierung eines maximalen Durchsatzes;
b) rechts abbiegende TE haben Vorrecht vor links abbiegenden;
c) „First come – first serve"-Modus, d.h., keine Richtung ist gegenüber einer anderen priorisiert.

Eine erste Dekomposition der Aufgabenstellung ergibt drei, zunächst als parallel abarbeitbare angenommene Teilfunktionen, die für jede Richtung die Steuerung der ankommenden TE übernehmen. Bild 5.27 zeigt den ersten Grobentwurf des Gesamtsystems und Bild 5.28 die erste Verfeinerungsstufe einer Teilfunktion. Die Verfeinerung des Task 1 zeigt bereits dessen vollen Funktionsinhalt, der hier in einem Schritt und ohne zusätzliche Arbeiten in einem Spezifikationsteil erzeugt wurde.

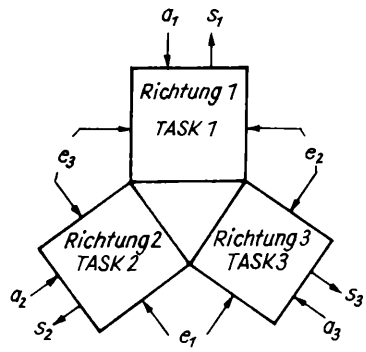


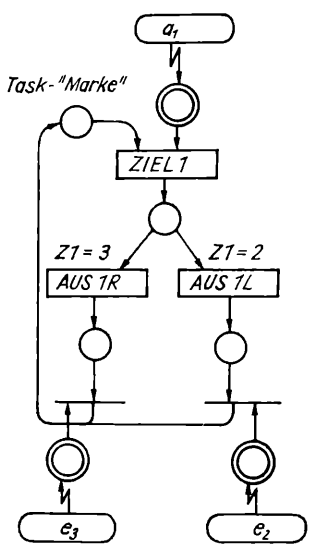
Bild 5.27
Funktioneller Grobentwurf des Beispiels

Bei komplexeren Aufgaben erfordert die Dekomposition und funktionelle Verfeinerung wesentlich mehr Aufwand und eine sorgfältige Systematik. Es bieten sich u. a. die Schritte

a) Objektmodellierung (z. B. auch mit Netz),
b) Hierarchieentwurf,
c) Datenflußentwurf

an (s. dazu [5.10]).

Bild 5.28 zeigt auch bereits die in den vorangegangenen Abschnitten erläuterte 2- Ebenen-Struktur, d. h., Verkopplungen mit der Prozeßebene sind nur über die Prozeduren und Interruptroutinen vorhanden. Der (funktionelle) Umfang der einzelnen Prozeduren wird durch die externe Signaleinkopplung und die Verzweigungsstruktur anhand



Interruptroutine: Annäherung einer TE aus Richtung 1

INPUT-Stelle (Pufferstrecke)

Prozedur: Identifikation der Zielrichtung Z1 (globale Variable) := Zielrichtung

Prozeduren: Stellen der Weiche W1 und Freigabe des Stoppunktes S1 über Ausgabe Signal s_1

Warten auf Entfernungsmeldung

INPUT - Stellen

Interruptroutinen: Entfernung einer TE in Richtung 2 (3)

Bild 5.28. Verfeinerung einer sequentiellen Teilfunktion

der globalen Variable Z1 bestimmt. Die Art und Weise der Realisierung der funktionellen Inhalte der Prozeduren ist zu diesem Zeitpunkt noch völlig ohne Belang.

In einem zweiten Schritt müssen nun die notwendigen Synchronisationsbeziehungen der Tasks untersucht und entworfen werden. Dies bedeutet konkret, daß die a priori angenommene Parallelarbeit (Nebenläufigkeit) der drei Teilkomponenten nun gewissen aufgabenspezifischen Einschränkungen unterworfen werden muß. Auch für diesen Schritt ist bei komplizierten Strukturen ein Modell des Steuerungsobjekts in Form eines Netzes sehr nützlich, während sich für das angegebene Beispiel die parallel ausführbaren Aktionsfolgen relativ leicht überschauen lassen. Man erkennt, daß alle links abbiegenden TE untereinander und die Links-Rechts-Paare (1/3), (2/1), und (3/2) die Kreuzung nur exklusiv benutzen dürfen, während alle rechts abbiegenden TE untereinander sowie die übrigen Links-Rechts-Paare die Kreuzung parallel passieren können. Eine solche Untersuchung ist nicht nur bezüglich der Forderung nach maximalem Durchsatz wichtig, sondern gibt auch die Basis für die Darstellung einer kausalen, d. h. zeitinvarianten Steuerung.

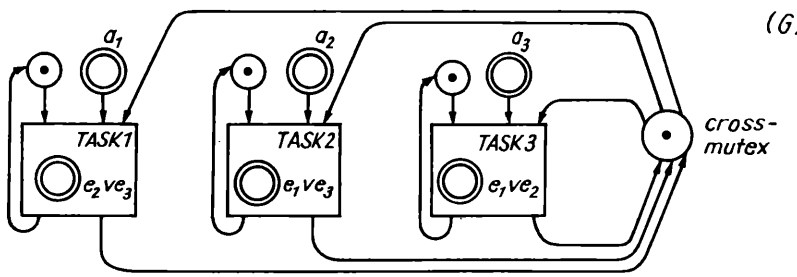


Bild 5.29. Synchronisation der Task durch globale Exklusivität

Die im Bild 5.29 gezeigte einfachste Synchronisationsvariante der Exklusivität der gesamten Kreuzung für jeden einzelnen Task widerspricht demnach nicht nur der Forderung a, sondern verdeckt auch die tatsächlichen kausalen Zusammenhänge. Die unzweckmäßige Vergröberung entsteht offensichtlich durch die Beschreibung der gesamten Kreuzung als eine einzige unteilbare Ressource (Stelle cross–mutex) im AKN. Eine mögliche Verfeinerung wäre die Zuordnung von bestimmten Kreuzungsteilen zu unterschiedlichen Ressourcen. Dafür bieten sich die vier Kreuzungspunkte K_0 bis K_4 geradezu an. Eine in Richtung i rechts abbiegende TE benötigt dann jeweils eine Ressource K_i und die links abbiegenden zusätzlich zu K_i noch die Ressource K_0. Die Realisierung im AKN eines Tasks ergibt sich durch die Zunahme der jeweils notwendigen Ressourcen in Form von markierten Stellen. Dieser zweite Verfeinerungsschritt ist noch nicht die endgültige Lösung, da bezüglich Forderung a noch Reserven vorhanden sind und Forderung b überhaupt nicht betrachtet wurde. Die genannte Reserve im Durchsatz ergibt sich aus der Freigabe der Ressource K_0, die erst gemeinsam mit der zweiten Ressource K_i erfolgt, also erst einige Zeit nach ihrer eigentlichen Benutzung. Durch Einfügen eines zusätzlichen Signals e_0 und einer weiteren Transition im AKN (Bild 5.30) ist das Problem lösbar.

Forderung b ist nur relevant, bei den bereits weiter oben angeführten zueinander exklusiven Links-Rechts-Paaren, da nur hier eine Konfliktsituation zwischen zwei unterschiedlich abbiegenden Transporteinheiten auftreten kann. Dieser Konflikt kann auch dann noch eintreten, wenn man das absolut gleichzeitige Eintreffen zweier TE vernachlässigt, da zugelassen ist, daß in den Pufferstrecken mehrere TE stehen können. So kann es durchaus zu einem Fall kommen, bei dem die Stelle I_{12} (Bild 5.30) im Task 1 und die entsprechende Stelle (I_{32}) im Task 3 markiert sind und außerdem Z1 = Z3 = 2 gilt. Die

Konfliktlösung mittels der Pufferstellen BR1 bis 3 (Nulltest über Inhibitorkante) ist im Bild 5.30 bereits enthalten. Erfolgt jetzt während der aufeinanderfolgenden Weiterleitung nach links mehrerer (im Puffer stehender) TE aus Richtung 1 eine Anmeldung einer TE aus Richtung 3 nach rechts, so wird die Abarbeitung der Linksabbieger über BR3 gestoppt und zunächst der Rechtsabbieger weitergeleitet. Voraussetzung bei dieser Herangehensweise ist allerdings, daß für die als parallel angenommenen Tasks auch tatsächlich je ein Aktionsträger (Rechner) zur Verfügung steht (Simultanarbeit). Eine solche

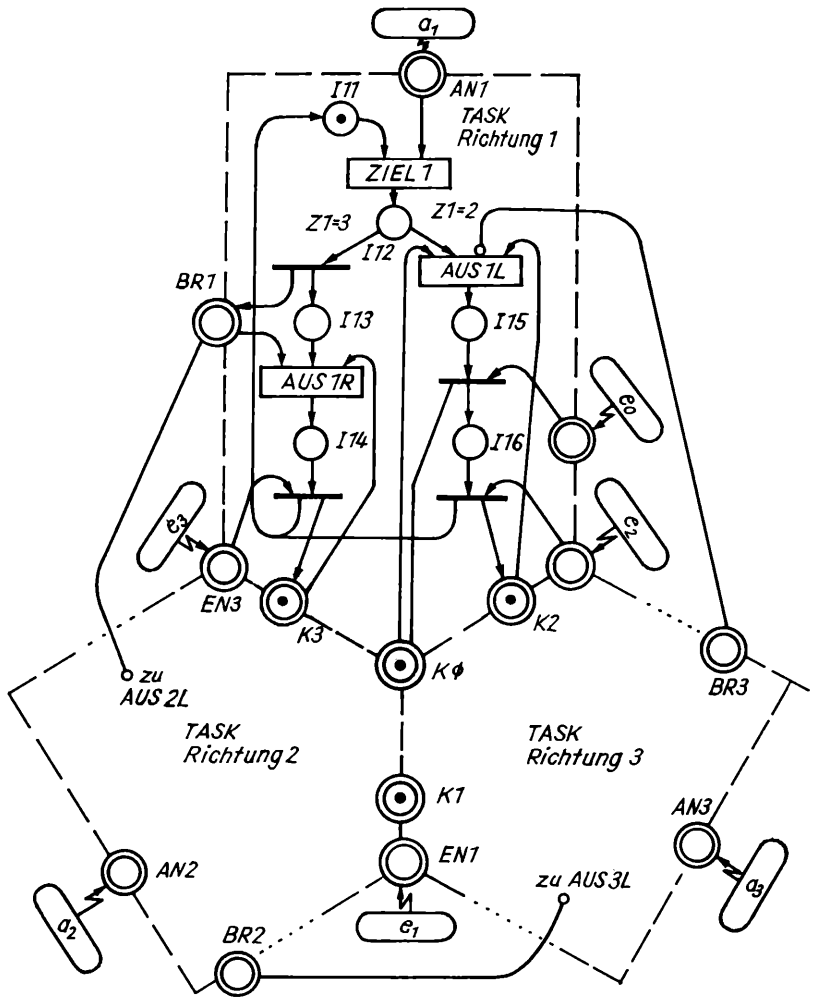


Bild 5.30. Problemlösung für simultan arbeitende Aktionsträger

implementierungsbezogene Frage muß zum Entwurfsanfang offengehalten werden, wird dann aber in einer bestimmten Stufe des Entwurfs endgültig geklärt (z.B. bei Prioritätsfragen). Die Lösung unter Berücksichtigung der zusätzlichen Forderungen b und c kann demnach für eine Multitaskvariante anders geartet sein als bei Simultanarbeit, was in der zusätzlichen Ressource Rechner begründet ist. Betrachtet man zunächst die Forderung c, so ist zu ihrer Erfüllung eine „faire" Vergabe von Prioritäten an die einzelnen Tasks zu realisieren (rollender Vergabemodus). Um das Netzmodell nicht unnötig zu verkompli-

zieren, wird diese Information in der Sprachnotation des Netzes angegeben. Außerdem ist es jetzt notwendig, auch die Unterbrechbarkeit der einzelnen Task zu betrachten, da ohne diese ein Multitasking nicht möglich ist. Jeder Task besitzt zwei Möglichkeiten der Unterbrechbarkeit:

a) innerhalb seiner Prozeduren, was anwenderspezifisch festgelegt werden kann durch die Deklaration der Prozedur als PROCESS (nicht unterbrechbar) oder ACTION (unterbrechbar), und
b) innerhalb des „Netz-Steuerprogramms", d. h. also, wenn sich die „Taskmarke" auf einer internen Stelle befindet.

Damit ist allerdings Forderung b immer noch nicht erfüllt, da es jetzt in dem oben angegebenen Beispielfall zwar zu einer Unterbrechung durch die TE-Anmeldung aus Richtung 3 kommt (Markierung von AN3), was aber nicht zwangsläufig auch zu einer sofortigen Ausführung des Task 3 führen muß, da durch die rollende Priorität gerade Task 1 bevorteilt sein kann. Das gewünschte Verhalten kann durch das Einrichten einer zweiten, höher priorisierten Ebene erreicht werden, in der die Identifikation der ankommendenTE vorgenommen wird. Dadurch zerfällt die bisherige Taskstruktur in zwei Teile. Bild 5.31

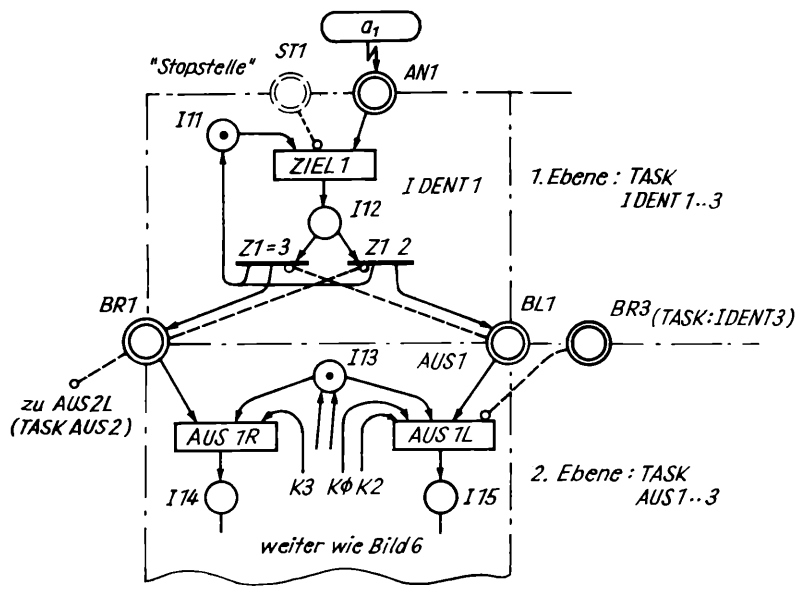


Bild 5.31. Taskstruktur einer Teilfunktion bei quasiparallel arbeitenden Aktionsträgern

zeigt die neue Netzstruktur, wobei der ehemalige Task 1 durch die Task IDENT1 und AUS1 ersetzt wurde. Jetzt wird bei Annäherung einer rechts abbiegenden TE aus Richtung 3 nicht nur AN3 markiert, sondern durch die immer höhere Priorität von IDENT3 gegenüber AUS1 (verschiedene Ebenen) kommt es zur Identifikation der TE und damit zur Markierung von Stelle BR3, vorausgesetzt natürlich, daß die Pufferstrecke 3 leer war. Damit ist, unabhängig vom momentanen Stand der Prioritäten von AUS1 und AUS3, die Prozedur AUS1L (und damit der Task AUS1) blockiert, und es kann AUS3 abgearbeitet werden. Die von den Pufferstellen BRi bzw. BLi ausgehenden Inhibitorkanten (Nulltest) bewirken das Sperren der Identifikation, wenn eine TE mit anderem Richtungswunsch als die in der Pufferstrecke schon vorhandenen eintrifft.

Über die angegebene Aufgabenstellung hinaus lassen sich nun noch Erweiterungen realisieren, die sich im Laufe des Entwurfsprozesses ergeben. So erscheint es beispielsweise sinnvoll, die Pufferstrecken durch zusätzliche Signale p_i zu überwachen, um ein unnormales „Ansammeln" von TE zu vermeiden. Eine solche Situation kann z.B. eintreten, wenn durch ungünstiges wechselseitiges Befahren der Kreuzung von zwei Richtungen die dritte Richtung über einen längeren Zeitraum keine Konzession erhält, weil immer eine notwendige Ressource fehlt (sog. Livelock). Durch ein solches Signal p_i soll über eine zusätzliche, entsprechend hoch priorisierte Task diese Richtung für eine ge-

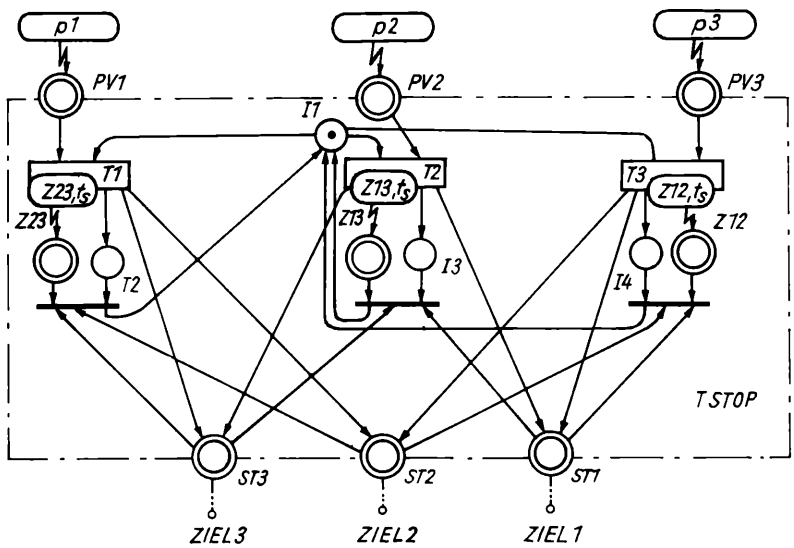


Bild 5.32. Task zur Verhinderung von Livelocks

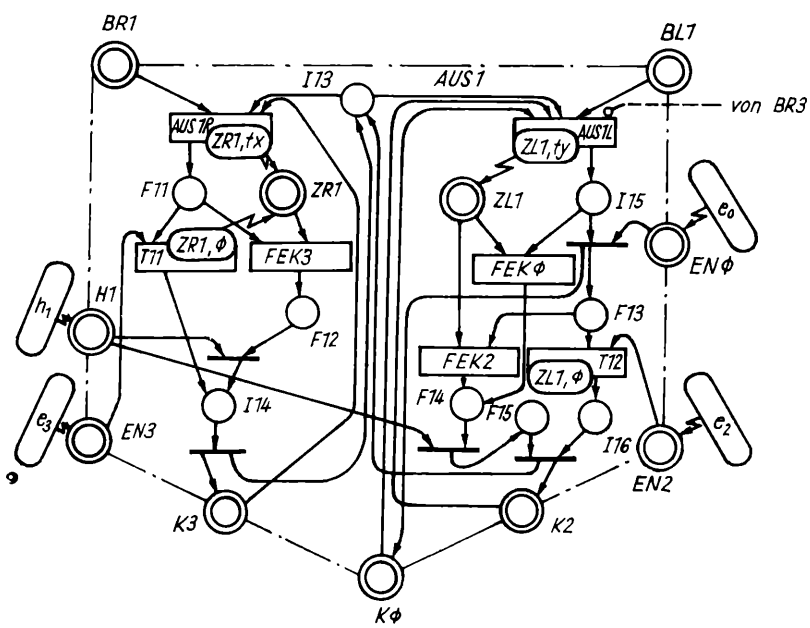


Bild 5.33. Erweiterung des Tasks AUS1 um eine Fehlerüberwachung

wisse Zeit die Kreuzung allein benutzen dürfen. Die Realisierung im AKN wird im Bild 5.32 gezeigt. Dabei bedeutet das Oval mit der ausgehenden Blitzsymbolkante in einigen Transitionen die Verwendung eines Systemmakros (DIRS *d*elayed *i*nterrupt *r*equest *s*ubroutine) zur verzögerten Markierung einer Ausgangsstelle, d.h. in dem konkreten Fall der Stellen Zij. Die Indizes i, j stehen dabei für die beiden Richtungen, die über die Teststellen STi bzw. STj für eine gewisse Zeit gesperrt werden.

Eine letzte Ergänzung sei dem Havariefall einer TE innerhalb des Kreuzungsbereichs gewidmet. Tritt eine solche Situation ein, soll eine hier nicht näher betrachtete Fehlerreaktion (Meldung) in Form von Prozeduren SEKi abgearbeitet werden. Die Überführung in den normalen Betriebszustand erfolge dabei über Handstellsignale h_i. Eine entsprechende Netzkonstruktion zur Überwachung im Task AUS1 wird in Bild 5.33 gezeigt. Dabei wird eine zeitliche Überwachung realisiert, d.h., wenn nach einer bestimmten Verzögerungszeit die Entfernungsmeldung e_i nicht erfolgt, werden die Stellen ZRi bzw. ZLi über die bereits erwähnten Systemmakros (DIRS) markiert, und die Fehlerprozedur erhält Konzession. Trifft das Signal e_i vor Ablauf der Verzögerungszeit ein, so wird die geplante Markierung über ein weiteres Systemmakro (DIRSO) zurückgenommen.

Eine besondere Fehlermaßnahme ist nicht notwendig, da die beteiligten Ressourcen K_i im Havariefall nicht freigegeben werden. Die Implementierung des Entwurfsmodells mit Hilfe einer speziell dafür geschaffenen Entwicklungssoftware wird im Abschnitt 5.5.4. beschrieben.

5.5. Struktureller und technischer Entwurf

5.5.1. Ermittlung der logischen Gleichungen aus dem PAG

Die Realisierung der mit den PAG entworfenen Steuerungsfunktion erfolgt als endlicher binärer Automat. Dadurch ergeben sich drei wesentliche Arbeitsschritte:

- Festlegung der Steuerungszustände im PAG;
- Festlegung der Kodierung der Steuerungszustände;
- Erfassung des Inhalts der PAG mit Hilfe Boolescher Gleichungen:
  Zustandsüberführungen,
  Ausgangslogik.

Aus der Vielzahl der Lösungsmöglichkeiten wird in [5.13] ein auf theoretischen Erkenntnissen und praktischen Erfordernissen und Erfahrungen beruhender Algorithmus ausführlich angegeben:

- Zuordnung der Steuerungszustände Z_i zu den Aktivitätenkästchen Y_j des PAG (dadurch Unterscheidung gleicher Aktivitäten im PAG).
- Mit Ausnahme des Anfangs- oder Ruhezustands Z_0 eines PAG sind die Zustände 1-aus-n-kodiert (je Zustand ein Flipflop).
- Der Anfangs- oder Ruhezustand Z_0 eines PAG wird unter dem Gesichtspunkt der Initialisierung beim elektrischen Zuschalten der Steuerung an das Netz eingenommen, wenn alle anderen Zustände des PAG nicht markiert (gesetzt) sind.
- Das Verfahren der Bestimmung der Booleschen Gleichungen ist in einer ersten Variante auf ein Betriebssystem der Steuerungstechnik zugeschnitten, das den Programmspeicherinhalt zyklisch vollständig abarbeitet (polling). Die Möglichkeit der Programmierung von Sprüngen in Abhängigkeit logischer Bedingungen (Steuerungssystem 600) kann dabei intuitiv genutzt werden.

In einer zweiten Variante der Bestimmung Boolescher Gleichungen nutzt man die Möglichkeit von Programmsprüngen in Steuerungen der Generation 600 gezielt zur schnelleren Abarbeitung des gesamten Gleichungssystems (Faktor 2 bis 3).

Verfahren der Zustandsgleichungen

Der inhaltliche Grundgedanke liegt darin, je Zustand z_i eine Gleichung zu schreiben. Da sie das Speicherverhalten mit realisieren muß, hat sie drei Bestandteile:

$$z_i := \text{Setzterm} + \text{Halten} \times \overline{\text{Rücksetzen}}$$

$$z_i := \bigvee_{\forall j \neq i} z_j h_{ji} + z_i \left(\bigwedge_{\forall k \neq i,0} \overline{z_k}\right) \overline{h_{i0}} \quad \text{für} \quad \forall i \neq 0 \tag{5.10}$$

$$z_0 = \bigwedge_{\forall i \neq \emptyset} \overline{z_i} \quad \text{für} \quad z_0; \tag{5.11}$$

j Index der Vorzustände zum Zustand z_i
k Index der Folgezustände zum Zustand z_i
h_{ji} logische Bedingung für den Zustandsübergang $z_j \rightarrow z_i$ (entspricht dem Weg im PAG von z_j nach z_i).

Diese Gleichungsform wird aus Gründen der sicheren Vermeidung von Fehlern infolge der seriellen Abarbeitung des Gleichungssystems und eines geringen Programmspeicherplatzbedarfs zur Erfassung des Inhalts von PAG empfohlen, die selbständig (nicht gekoppelt mit anderen Graphen) arbeiten und $i \lesssim 4$ Zustände besitzen.
Spezialfall für $i = 2$ (hier gilt $z_0 = \overline{z_1}$):

$$z_1 = \overline{z_1} h_{01} + z_1 \overline{h_{10}}$$

oder

$$z_1 = \overline{z_1} h_{01} + z_1 h_{11}. \tag{5.12}$$

Modifiziertes Verfahren der Zustandsgleichungen

Diese Modifikation ist universell anwendbar. Gegenüber dem ursprünglichen Verfahren wird dieser Effekt dadurch erzielt, daß die Setzterme der Zustände als selbständige Zwischensignale (Adressen temporärer Merker der Steuerungstechnik) berechnet und danach erst die Zustandsgleichungen bestimmt werden.

Der Einsatz empfiehlt sich für

selbständige PAG mit $i \gtrsim 4$ Zuständen,
alle anderen PAG mit $i > 2$ Zuständen.

Reihenfolge der Gleichungen:

1. Setzterme

$$s_i = \bigvee_{\forall i \neq i} z_j h_{ji} \quad \forall i \tag{5.13}$$

2. Rücksetzterm (einer je PAG)

$$r = \bigvee s_i \quad \forall i \tag{5.14}$$

3. Zustandsgleichungen

$$z_i := s_i + z_i\bar{r} \qquad \forall i \neq 0 \tag{5.15}$$

$$z_0 = \bigvee \bar{z}_i \qquad \forall i \neq 0$$

4. Ausgangslogik.

Hervorzuheben ist, daß nur noch die Setzterme und die Ausgangslogik aufgabenspezifisch sind. Der Rest könnte formal ergänzt werden. Verwiesen sei auch auf die Möglichkeit, die Gleichung für z_0 (Ruhezustand) zu eliminieren, indem an den erforderlichen Stellen die Gleichung für z_0 explizit eingesetzt wird (für den Zustand z_0 wird dann kein Signal mehr gebildet).

5.5.2. Ermittlung der logischen Gleichungen aus dem Steuernetz

Das beschriebene Notierungsverfahren für Programmablaufgraphen (Zustandsgleichung) wurde zur Steuernetzumsetzung folgendermaßen erweitert:

1. Bildung aller Setz-Rücksetz-Terme eines Netzes
   Alle Transitionen mit den zugehörigen Schaltbedingungen und Vorplätzen werden als Terme (Merker) erfaßt:

$$t_i = f_i \wedge s_j$$
$$\forall s_j;$$

f_j Schaltbedingung der Transition T_i
s_i Vorplätze von der Transition T_i.

2. Bildung der Zustandsgleichung
   Für jeden Platz des Netzes werden die Vor- und Folgetransitionen erfaßt und in eine Boolesche Gleichung eingearbeitet:

$$\text{Platz} = \bigvee \text{Setzterme} + \text{Platz} \times \bigwedge \overline{\text{Rücksetzterme}}$$

$$s_k = \bigvee t_s + s_k \times \bigwedge t_r;$$

k Index des entsprechenden Platzes
s Index der Vortransitionen des Platzes
r Index der Folgetransitionen des Platzes.

3. Anfangsmarkierung
   In Abhängigkeit von der eingesetzten Steuerung kann die Anfangsmarkierung unterschiedlich realisiert werden. In speicherprogrammierbaren Steuerungen mit ausschließlich Boolescher Logik kann die Anfangsmarkierung durch Erweiterung der Zustandsgleichungen der im Anfangszustand markierten Plätze realisiert werden:

$$s_j = \bigvee t_s + \bar{m} + s_j \times \bigwedge \bar{t}_r, \qquad \forall s_j \in m_0;$$

m Merker für die Anfangsmarkierung, der beim erstmaligen Programmdurchlauf den logischen Wert 0 besitzt und erst am Programmende auf den Wert 1 zu setzen ist.

Dieses auf dem Zustandsbegriff (Plätze = Zuständen) basierende Verfahren kann bei der Notation von Netzen ohne Parallelarbeit durch die für Programmablaufgraphen entwickelten Verfahren ersetzt werden.

5.5.3. Implementierung des AKN-Entwurfs mit dem Softwareentwurfssystem PSI 1520

An die im vorangegangenen Kapitel skizzierte Entwurfsphase eines AKN schließen sich je eine Analyse-, Test- und Implementierungsphase an, die durch das System PSI 1520 (Petri-Netz-Simulation und Implementierung für Zielrechner der Generation K1520) maximal unterstützt werden. Das System wurde zur Softwareerstellung für kleinere bis mittlere Automatisierungsvorhaben entwickelt, die sich insbesondere durch kritisches Echtzeitverhalten, hohen Parallelitätsgrad und starke Koppelbeziehungen zwischen einzelnen Teilaufgaben auszeichnen und somit auch günstig für die softwaremäßige Realisierung der Koordinierungssteuerung von Industrierobotern einsetzbar ist. Das System arbeitet mit dem Beschreibungsmittel AKN und benötigt als gerätetechnische Grundlage einen Bürocomputer BC A5120 mit dem Betriebssystem UDOS.

In einem ersten Schritt ist eine AKN-gerechte Eingabeform für das Entwurfssystem PSI 1520 zu erstellen. Dies geschieht entweder in einer Sprachversion oder in einer gemischt grafisch-alphanumerischen Version. Unabhängig von der Eingabeversion gibt es immer einen Deklarationsteil, der die im Netz bezeichneten Stellen als Variable bestimmten Typs definiert. Es wird dabei unterschieden zwischen taskinternen Stellen (INTERNT), die nur der sequentiellen Verkopplung der Prozeduren im Task dienen; sicheren Intertaskkoppelstellen (SAVE); beschränkten Intertaskkoppelstellen (BOUND), die für einen Systemeinsprung (CALLOS) mit angebbarer Zielebenenpriorität generierbar sind sowie den Nulltest ermöglichen; externen Intertaskkoppelstellen (EXTERN), die der Kommunikation zwischen Task verschiedener Rechner dienen, und Stellen mit Kontakt zur Prozeßperipherie, die sich in zeitbewertete Stellen (TIMED) und Interrupteingabestellen (INPUT) aufteilen. Außer den Peripheriestellen können hier alle anderen auch mit einer Anfangsmarkierung versehen werden.

Anschließend werden alle Tasks ebenenweise notiert oder grafisch auf den Bildschirm „gezeichnet". Dabei kann für jede Ebene unter dem Schlüsselwort PMODE eine Prioritätsverteilungsstrategie aus den Möglichkeiten FIX (feste Prioritätsverteilung), VAR (feste, aber dynamisch änderbare Prioritätsverteilung) und ROLL (rollende Prioritätsverteilung) ausgewählt werden. Die der Ebene zugewiesene eigene Priorität entspricht einer festen Prioritätsverteilung unter allen angegebenen Ebenen. Die Tasks (TASK) werden nach einer möglichen Angabe ihrer Anfangspriorität in der jeweiligen Ebene transitionsweise (TRANS) notiert („gezeichnet"), wobei in der grafischen Variante nur ein drei Zeichen langer Name vergeben wird, über den dann in einer alphanumerischen Eingabezeile die weiteren Eigenschaften (ACTION, PROCESS usw., s. oben) definiert werden. In der Sprachversion müssen dann zusätzlich alle Eingangs- (IN) und Ausgangsstellen (OUT) sowie evtl. vorhandene Inhibitoreingänge (INHIB) oder Formeln (COND) als zusätzliche Schaltbedingung angegeben werden.

Bild 5.34 zeigt Ausschnitte aus der Sprachversion für das zuletzt angegebene Netzmodell der Bilder 5.31, 5.32 und 5.33. Mit der Fertigstellung des Entwurfs der Koordinierungssteuerung in Form eines entsprechend beschrifteten Ablaufkontrollnetzes kann mit dem Entwurf der Aktionssteuerung (Prozeduren, Interruptroutinen) begonnen werden. Auch dafür bietet das System PSI in Form des Programmpakets PSI.PPC (PSI-*p*rocedure-*pre*compiler) eine gute Unterstützung. Es besteht aber auch die Möglichkeit, die entworfene Koordinierungssteuerung zu analysieren und mit abzuschätzenden Prozedurlaufzeiten auf seine Echtzeitfähigkeit auszutesten, ohne eine einzige Prozedurzeile programmieren zu müssen.

```
LEVEL AUSGABE PRIORITY:=3 PMODE=ROLL;
  TASK AUS1;
    (* AUSGABE-PROZEDUREN *)
    (IN I13, BR1, K3 OUT F11)
     ACTION AUS1R STACK:=8 PUSH=COMPLETE;
    (IN I13, BL1, KO, K2 OUT I15 INHIB BR3)
     ACTION AUS1L STACK:=8 PUSH=COMPLETE;
    (* FEHLER-PROZEDUREN *)
    (IN I15, ZL1    OUT F11)
     ACTION FEKO STACK:=6 PUSH=SIMPLE;
    (IN F13, ZL1   OUT F14)
     ACTION FEK2 STACK:=6 PUSH=SIMPLE;
    (IN F11, ZR1   OUT F12)
     ACTION FEK3 STACK:=6 PUSH=SIMPLE;
    (* TIME OUT-UEBERWACHUNG *)
    (IN F11, EN3   OUT I14) PROCESS T11;
    (IN F13, EN2   OUT I16) PROCESS T12;
    (* SCHALTTRANSITIONEN *)
    (IN F12, H1    OUT I14);
    (IN F14        OUT I13, K3);
    (IN F15, EN0   OUT K0, F13);
    (IN F14, H1    OUT F15);
    (IN F15, I16   OUT I13, K2);
ENDTASK;
  TASK AUS2; ... ENDTASK;
  TASK AUS3;
    :
    (IN F35, I36   OUT I33, K1);
  ENDTASK
ENDLEVEL.
NET DREIWEG;
  (* STELLENDEKLARATION *)
  INTERN  I1:=1, I11:=1, I13:=1, I21:=1, ... , I33:=1
          (* TASK-ANFANGSMARKIERUNG *),
          I2, I3, I12, I14, I15, I16, I22, ... , I36,
          F11, ... , F15, F21, ... , F35;
  SAVE    K0:=1, K1:=1, K2:=1, K3:=1 (* KREUZUNGSPUNKTE *)
  BOUND   ST , ST2, ST3, (* TASKSTOP-TEST *)
          BR1, BL1, BR2, BL2, BR3, BL3;
  INPUT   AN1, AN2, AN3, (* ANNAEHERUNG *)
          EN0, EN1, EN2, (* ENTFERNUNG *)
          PV1, PV2, PV3, (* PUFFER VOLL *)
          H1, H2, H3 (*HANDSIGNALE *);
  TIMED   Z23, Z12, Z13, (* TASKSTOP-ZEIT *)
          ZR1, ZL1, ZR2, ZL2, ZR3, ZL3 (* TIME OUT-UEBERWACHUNG *);
  (* EBENENDEFINITIONEN *)
  LEVEL TSTOP PRIORITY:=1 PMODE=FIX (* TASKSTOP BEI PUFFER VOLL *);
    TASK TSTOP;
      (* STOPZEIT-PROZESSE *)
```

Bild 5.34

```
    (IN PV1, I1  OUT ST3, I2, ST2) PROCESS T1
                                   (* MARKIERT VERZOEGERT Z23 *);
    (IN PV2, I1  OUT ST1, I3, ST2) PROCESS T2;
    (IN PV3, I1  OUT ST1, I4, ST2) PROCESS T3;

    (* SCHALTTRANSITIONEN *)
    (IN I2, Z23, ST2, ST3  OUT I1);
    (IN I3, Z12, ST1, ST2  OUT I1);
    (IN I4, Z13, ST1, ST3  OUT I1);

  ENDTASK
ENDLEVEL;
LEVEL IDENTIFY PRIORITY:=2 PMODE=ROLL (* ZEITIDENTIFIKATION *);
  TASK IDENT1;
    (* ZIELBESTIMMUNG *)
    (IN AN1, I11  OUT I12 INHIB ST1)
     ACTION ZIEL1 STACK:=10 PUSH=SIMPLE;
    (* ENTSCHEIDUNGSTRANSITIONEN *)
    (IN I12  OUT I11, BR1  INHIB BL1) COND Z1=3;
    (IN I12  OUT I11, BL1  INHIB BR1) COND Z1=2;
  ENDTASK;
  TASK IDENT2; ... ENDTASK;
  TASK IDENT3; ... ENDTASK
ENDLEVEL;
```

Bild 5.34. PSI-Sprachversion des Transportbeispiels

Bei der Analyse wird die Möglichkeit des Netzbeschreibungsmittels ausgenutzt, daß bestimmte für die Steuerung relevante Netzeigenschaften analytisch untersucht werden können. Herausragendste Eigenschaft ist dabei die sog. Netzlebendigkeit, die für die Steuerung garantiert, daß kein Verklemmungszustand durch die Steuerung selbst verursacht werden kann. Voraussetzung für die Arbeit mit dem Analyseprogrammpaket PSI.ANALYSIS ist der vom Netzcompiler PSI.COM aus der PSI-Sprachform übersetzte Zwischenkode. Die Sprachversion wird dabei entweder über einen normalen Quelltexteditor des Trägerentwicklungssystems erstellt, oder, wenn der Entwurf am Bildschirm erfolgt, mittels Programmpaket PSI.GRAPHIC erzeugt. Sind die Analyseergebnisse befriedigend, erfolgt mit dem Kodegenerator PSI.CG die weitere Verarbeitung des Zwischenkodes. Wird eine Weiterarbeit auf Simulationsbasis gewünscht, erzeugt PSI-CG eine Datei NSIM, die aus folgenden Teilstücken besteht:

- NSB (*n*et *s*tructure *b*lock) – dieses Stück bildet den problemabhängigen Teil des PSI-Laufzeitsystems in einer besonders komprimiert aufbereiteten Form der Netztopologie;
- DSA (*d*ynamic *s*torage *a*llocation) – Arbeitsspeicher des PSI-Laufzeitsystems, enthält auch den Stack der Aktionsprozeduren;
- TSP (*t*ime *s*imulation of *p*rocedures) – schafft die Möglichkeit einer variablen zeitlichen Simulation aller Aktionen und Prozesse.

Diese Datei NSIM wird mit dem PSI-Echtzeitkern PSI.RTN (PSI-*r*eal-*t*ime-*n*ucleus) und dem Testprogramm PSI.DEBUG im Objektformat gebunden und über PSI.DEBUG gestartet. Dieser Debugger bietet in Form von Traceprotokollen einen Überblick über die zeitliche Reihenfolge aller Systemereignisse und die Systemauslastung. Dazu arbeitet

PSI.DEBUG in zwei Moden: einem Usermode (dann ist das PSI-Laufzeitsystem PSI.RTN aktiv) und einem Systemmode, während dessen Aktivzeit die Echtzeituhr gestoppt wird. Auf diese Weise sind ähnlich einem Emulationssystem zu beliebigen Zeitpunkten Eingriffe in den Ablauf vornehmbar (oder planbar), ohne die Echtzeit zu verfälschen. Voraussetzung einer solchen Arbeitsweise ist allerdings die Identität des Zielprozessors mit dem des Entwicklungssystems, oder es müssen spezielle Hardwareemulatoren gebaut werden. PSI.DEBUG bietet u.a. folgende Möglichkeiten:

- Einstellen und Verändern der Prozedurlaufzeiten,
- Setzen von Unterbrechungspunkten in Prozeduren,
- zyklisches und azyklisches (verzögert, stochastisch) Markieren von Peripheriestellen,
- Einstellen und Verändern der Werte von globalen Variablen,
- Einstellen und Verändern von Prioritätsverteilungsstrategien und Prioritätsrangfolgen,
- Ein- und Ausschalten der Tracefunktion,
- Auflistung der momentan aktuellen Markierungen.

Für das oben angeführte Beispiel ist ein kurzer Ausschnitt eines Traceprotokolls in Tafel 5.2 angegeben, der eine Konfliktsituation mit Lösung „rechts vor links" der Richtungen 3 und 1 und eine Parallelfahrt (1 links, 2 rechts) zeigt.

Tafel 5.2. Ausschnitt aus einem Trace-Protokoll

PSI.DEBUG: TRACE-PROTOKOLL: NET DREIWEG
CLOCK CYCLE: 100 ms (not printed)

Time (MIN:SEK:USEK)	Aktivity	Name	Place for comment
00:00:000	SYS		Dyn. Stopzustand
00:02:104	I	A1	Interrupt a_1
00:02:104	A	ZIEL1	Start Proc. ZIEL1
00:02:302	I	A3	
00:02:303	A(R)	ZIEL1	Restart Proc. ZIEL1
00:02:523	A	ZIEL3	●
00:02:548	A	AUS3R	▲
00:03:648	SYS		↕ Fahrt (3,2)
00:07:100	I	F3	●
00:07:101	Aü	AUS1L	▲
00:08:201	SYS		
00:10:050	I	A2	Fahrt (1,2)
00:10:050	A	ZIEL2	●
00:10:270	A	AUS2R	▲
00:11:370	SYS		↕ Fahrt (2,1)
00:13:700	I	E2	●
00:13:701	SYS		▼
00:14:200	I	E1	●
00:14:201	SYS		

Durch eine stochastische Belegung der Anforderungssignale a_i unter einer wählbaren Verteilung (Gleich-, Gauß- oder Poisson-Verteilung) lassen sich auch bedientheoretische Aussagen bezüglich Reaktionszeiten und Rechnerauslastung treffen.

Erst nach erfolgter Zeitsimulation und evtl. notwendigen Präzisierungen (iterative Schritte) in der Netzstruktur sollte der Entwurf der Aktionssteuerung beginnen. Über

die im Debugging gewonnenen Erkenntnisse ist ein genaues Laufzeitmaximum aller Prozeduren sowie deren Arbeit mit den globalen Variablen der Koordinierungsebene vorgegeben. Natürlich ist auch ein umgekehrtes Vorgehen, beginnend mit dem Entwurf der Prozeduren (z.B. bei Standardprogrammen), denkbar.

Für die Gestaltung aller Programme in der Aktionssteuerung stehen PSI-Systemmakros zur Verfügung, die durch den Vorübersetzer PSI.PPS so verarbeitet werden, daß Assemblerquellprogramme entstehen. Dabei wird gleichzeitig die richtige Benutzung der Makros bezüglich des entworfenen Netzes, unter Benutzung der Datei NSB, getestet. Liegt noch kein Netz vor, werden bestimmte Randwerte im Dialog abgefragt.

Tafel 5.3. Systemmakros von PSI 1520

	Name	Parameter	Funktion
1	STRT	NAME	Prozedurbeginn für ACTION und PROCESS NAME
2	INTR	–	Beginn einer Interruptroutine
3	AEND	NAME	Prozedurende für ACTION NAME
4	PEND	NAME	Prozedurende für PROCESS NAME
5	IEND	–	Ende einer IR
6	DIRS	PLN,DEL	(delayed interrupt sequest subroutine) PLN (place name) vom Typ TIMED DEL (delag) ganzzahliges Vielfaches vom Grundtakt um DEL verzögertes Markieren von Stelle PLN über die PSI.RTN-Echtzeituhr (IR:EUHR), wenn DEL=∅ Austragen einer geplanten Markierung für PLN, wenn DEL=∅
7	PINC*	PLN	(placemarking increment) PLN vom Typ INPUT m (PLN): = m (PLN) + 1
8	PRES*	PLN	(placemarking reset) PLN vom Typ INPUT m (PLN): =∅
9	PMOD	LPN,MODE	(priority modification of level) LPN (level priority number) MODE = { FIX wenn ∅; VAR wenn 1; ROLL wenn 2 } Prioritätsvergabestrategie von Ebene LNR ergibt sich zu Mode
10	VAPA	LPN,SPOS, DPOS	(variable priority assigment) LPN (wie oben) SPOS=(source position) alte Prioritätsposition DPOS=(drain position) neue Prioritätsposition in LPN Task von SPOS wird auf DPOS eingeordnet, dazwischenliegende Positionen werden verschoben
11	LPIN	LPN	(level priority-sheduling initiatization), PLN (w.o.) einer Ebene LPN = ∅ Herstellen des PMODE-Initialzustandes aller Ebenen LNP = ∅

*) Makros dürfen nur in Interruptroutinen verwendet werden

Eine Aufstellung der bisherigen Systemmakros gibt Tafel 5.3 an. Werden Prozeduren in PLZ-SYS erstellt, so sind die Makros entweder in externen Assemblerprozeduren unterzubringen oder werden nach der PLZ-Compilierung in der Assemblerquelltextstufe eingefügt. Für das Entwurfsbeispiel sind in den Bildern 5.35 und 5.36 einige Prozeduren und Interruptroutinen auszugsweise angegeben. Sind alle Prozeduren und Interruptroutinen erstellt und getestet, wird eine Trennung zwischen Programmspeicher- und Datenspeicherteilen vorgenommen, die anschließend zweckmäßigerweise zusammengefaßt werden, um sie zu zwei Objektdateien für ROM und RAM zu übersetzen. Diese Dateien bilden dann das beim Debugging benutzte Teilstück TSP. Für die Realisierung auf einem

```
;TASK TSTOP:
          STRT      T1
    EXTERNAL        Z23
T1        DIRS      Z23 250 ;250 * 100 MS = 25 SEC VERZOEGERUNG
          ENDP      T1
;
;
;         .
;TASK IDENT1:
          STRT      ZIEL1
    EXTERNAL        Z1
ZIEL1     EI                    ;ERLAUBT INTERRUPT
;
;                   IDENTIFIKATION: ERGEBNIS IN <A>
;         .
          LD        (Z1), A     ;GLOBALE VARIABLE Z1
          ENDA      ZIEL1
;
;TASK AUS1:
          STRT      AUS1R
    EXTERNAL        ZR1
AUS1R
;         STELLBEFEHL: W1 NACH RECHTS
          EI
          WARTEN 0.5 SEC    WEICHENSTELLZEIT
          STOPSTELLE S1:=FREI
          WARTEN 0.5 SEC    PASSIERZEIT EINER TE > UNTEILBARE OPERATION
          STOPSTELLE S1:=BESETZT
          DIRS ZR1 40       ;4 SEC TIME OUT ZEIT
          ENDA      T1

          STRT      T11
    EXTERNAL        ZR1
T11       DIRS      ZR1 0       ;AUSTRAGEN DER IR-ANMELDUNG VON AUS1R
          ENDP      T11
;
;
```

Bild 5.35. Ausschnitte aus den Taskprogrammen

```
    EXTERNAL        AN1, AN2, AN3, EN0, EN1, EN2, EN3
SAMI1     INTR
          IN        PORT1
          RRA
          JRC       A2
          PINC      AN1
          ENDI
A2        RRA
          JRC       A3
          PINC      AN2
          ENDI
;
;
;         .
E2        RRA
          JRC       E3
          ENDI
E3        PINC      EN3
          ENDI
```

Bild 5.36. Ausschnitt aus dem Interruptprogramm

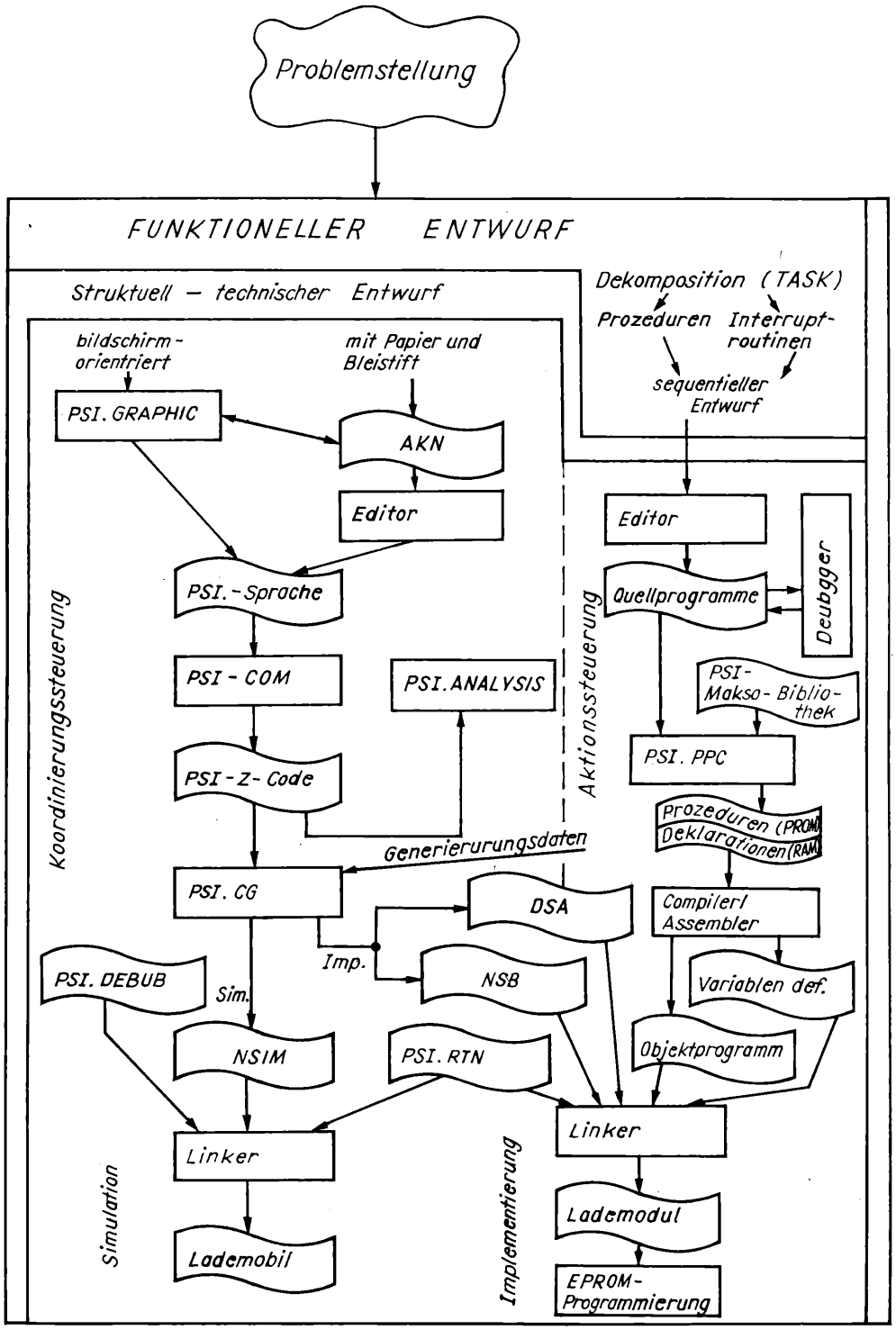


Bild 5.37. Entwurfsphasen und Bestandteile des Systems PSI

konkreten Zielsystem ist auch für die Strukturblock- und Arbeitsspeicherdatei (NSB und DSA) eine Aufspaltung mittels PSI.CG notwendig, da gegenüber der Testvariante auf dem Entwicklungssystem einige zielsystemabhängige Informationen benötigt werden. Insbesondere betrifft das die Speicheraufteilung (PROM für NSB und RAM für DSA) und die Generierung der Echtzeituhr, für die ein CTC-Baustein mit zwei koppelbaren Kanälen (Zeitgeber/Zähler) notwendig ist. Für diesen wird vom Kodegenerator die notwendige Information im Dialog zusammengestellt, wobei folgende Daten einzugeben sind:

- CTC-Interruptvektor-Tabelle (Adresse),
- CTC-Basis-Adresse (Kanal 0),
- Nummer des Zeitgeberkanals,
- Nummer des Zählerkanals,
- EUHR-Grundtakt (als Vielfaches von 10 ms).

Die weiter notwendige Generierung des Zielsystems (z.B. Initialisieren der Peripheriebausteine für externe Interruptsignale bzw. die Prozedur-Ein-/Ausgaben zur Prozeßperipherie) muß vom Anwender selbst vorgenommen werden. Dafür wird vom PSI-Laufzeitsystem während der Anlaufphase (vor der Uhr-Initialisierung und dem damit verbundenen „heißen" Start des Gesamtsystems) eine Aussprung- und Wiedereinsprungtabelle bereitgestellt. In diesem dort einfügbaren Programmstück ist es auch möglich, globale Variable zu initialisieren, die vorher von PSI-RTN nur rückgesetzt werden.

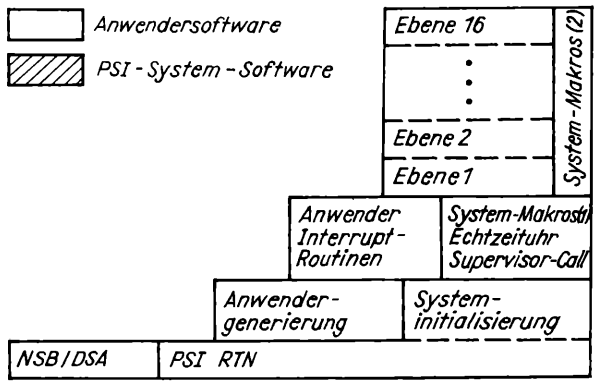


Bild 5.38
Struktur der mit PSI erzeugten Zielsoftware

Alle zu Objektdateien aufbereiteten Programme werden abschließend zu einem zielrechnerangepaßten Softwareendprodukt zusammengebunden und auf entsprechende Datenträger (EPROM-Speicherschaltkreise) übertragen.

Bild 5.37 zeigt nochmals den Ablauf und die Bestandteile des Systems PSI, während auf Bild 5.38 die Strukturierung der mit diesem System erstellten Software zu sehen ist.

6. Programmierung von Industrierobotern

6.1. Industrieroboter als programmierbare Systeme

Als wesentliches Kennzeichen der Industrieroboter ist ihre freie Programmierbarkeit anzusehen. Durch die Programmiermöglichkeit sind IR mit anderen programmierbaren Systemen vergleichbar, die als Kernstück einen Rechner (Prozessor, Automat) enthalten. Der Anwender formuliert die Aufgabe als Programm in einer durch das IR-System vorgeschriebenen Form (Eingabeform). Das IR-System setzt dieses Programm in Bewegungen und Aktionen des IR im technologischen Umfeld um.

Gegenüber der Programmierung von Rechnern für Datenverarbeitungsprobleme und für wissenschaftliche Berechnungen weist die IR-Programmierung eine Reihe von speziellen Problemen auf. Prinzipiell ist zwischen der externen und der internen Programmierung zu unterscheiden (vgl. die Gegenüberstellung innere und äußere Software im Abschn. 2.6.).

Die *externe Programmierung* dient zur Aufstellung eines Bewegungs- und Aktionsprogramms (Roboterprogramm) in einer für den Nutzer handhabbaren Form. Das IR-System ist dabei als eine „virtuelle" Maschine aufzufassen, die wie ein Rechner arbeitet (s. Abschn. 2.6.).

Die *interne Programmierung* dient zur Programmierung des im IR-System enthaltenen Rechners und erfolgt durch den Entwickler des Steuerungs- und Programmiersystems in einer Rechnerprogrammiersprache (z.B. MACRO oder PASCAL).

Spezielle Probleme der externen IR-Programmierung sind darin begründet, daß das Programmiersystem den Roboter in einer realen Umwelt steuern muß. Statusänderungen, die durch die IR-Aktionen hervorgerufen werden, können durch Sensorsysteme erkannt und zur Modifikation der Abarbeitung des Roboterprogramms herangezogen werden. Der Grad der Anpassungsfähigkeit ist vom Entwicklungsstand des IR-Systems abhängig (vgl. Robotergenerationen im Abschn.1.).

An folgendem Beispiel lassen sich die Grundelemente der IR-Programmierung erkennen:

Beispielaufgabe

Mehrere Werkstücke sollen von einem Förderband abgenommen und auf einer Palette abgelegt werden. Der Ablauf wird mit einer Meldung beendet, wenn keine Werkstücke mehr vorhanden sind bzw. die Palette voll belegt ist.

Lösung

⟨Definiere Position Förderband⟩
⟨Definiere Position Palette (Ablage 0)⟩
⟨Definiere Ablagendifferenz für Palette⟩
⟨Definiere Ablagenanzahl⟩
⟨Wiederhole folgende Anweisungen bis Endbedingung erfüllt⟩
⟨Bewege Arm zur Position Förderband⟩
⟨Schließe Greifer⟩
⟨Wenn kein Werkstück, setze Endbedingung⟩

⟨Palettenposition = Palettenposition + Ablagendifferenz⟩
⟨Ablagenanzahl = Ablagenanzahl − 1⟩
⟨Wenn Ablagenanzahl = 0, setze Endbedingung⟩
⟨Bewege Arm mit Werkstück zur aktuellen Palettenposition⟩
⟨Öffne Greifer⟩
⟨Zyklusende⟩
⟨Gebe Meldung an Bediener⟩
⟨Warte auf Eingabe⟩

6.2. Programmierverfahren für Industrieroboter

Bei der Programmierung von Industrierobotern werden folgende Verfahren unterschieden:

- manuelle Programmierung (Verbindungsprogrammierung),
- direktes Teach-in,
- indirektes Teach-in,
- textuelle Programmierung.

Diese Verfahren lassen sich z.T. weiter unterteilen. Entscheidend bei allen Programmierverfahren ist, ob für die Aufstellung des Programms der Manipulator eingesetzt wird oder nicht. In den Fällen, wo der Manipulator zur Programmerstellung nicht gebraucht wird, spricht man von einer Off-line-Programmierung. Die anderen Formen bezeichnet man mit On-line-Programmierung. Dabei wird der Roboter von Hand durch das Programm geführt oder gesteuert und so u. a. als Meßmaschine zur Ermittlung der benötigten Raumpunkte und Orientierungen eingesetzt. Die Off-line-Programmierung hat den Vorteil, daß der Manipulator nicht aus dem laufenden Prozeß herausgenommen werden muß, wenn ein neues Programm erstellt wird. Das Verfahren ist jedoch weitgehend auf textuelle Programmierung und eine explizite Erfassung und Beschreibung der geometrischen Umweltbezüge angewiesen. Da nicht alle Funktionen problemlos off-line programmiert werden können, treten verschiedene Mischformen von Programmierverfahren auf. Es sei darauf hingewiesen, daß die Bezeichnung der Verfahren nicht einheitlich ist. So heißt z. B. das direkte Teach-in auch Folgeprogrammierung. Eine Übersicht zu den verschiedenen Programmierverfahren gibt Bild 6.1.

Tafel 6.1 gibt dazu die bei der Programmierung jeweils auszuführenden Aktionen und typische Einsatzfälle für die einzelnen Programmierungsarten an.

6.2.1. Manuelle Programmierung (Verbindungsprogrammierung)

Einfache Handhabungsaufgaben (Einlegen, Beschicken von Werkzeugmaschinen) sind dadurch gekennzeichnet, daß wenige Positionen im Arbeitsraum sehr häufig und in fester Folge eingestellt werden sollen. Für solche Aufgaben sind einfache Steuerungen geeignet. Das Programm wird manuell in Form fester Verbindungen eingebracht (Verbindungsprogrammierung). Vom Hersteller solcher verbindungsprogrammierbaren Steuerungen werden alle notwendigen Baugruppen geliefert, die der Anwender selbst zusammenschalten muß. Die logische Steuerung kann auf pneumatischer, hydraulischer, mechanischer oder elektrischer Basis arbeiten. Die einzelnen Haltepunkte der einstellbaren Achsen lassen sich z. B. durch Anschläge realisieren und liegen daher als Positionen in Roboterkoordinaten vor. Die manuelle Programmierung ist einfach zu realisieren; für einen flexiblen Einsatz ist sie jedoch wenig geeignet, da der Umrüstungsaufwand relativ groß ist.

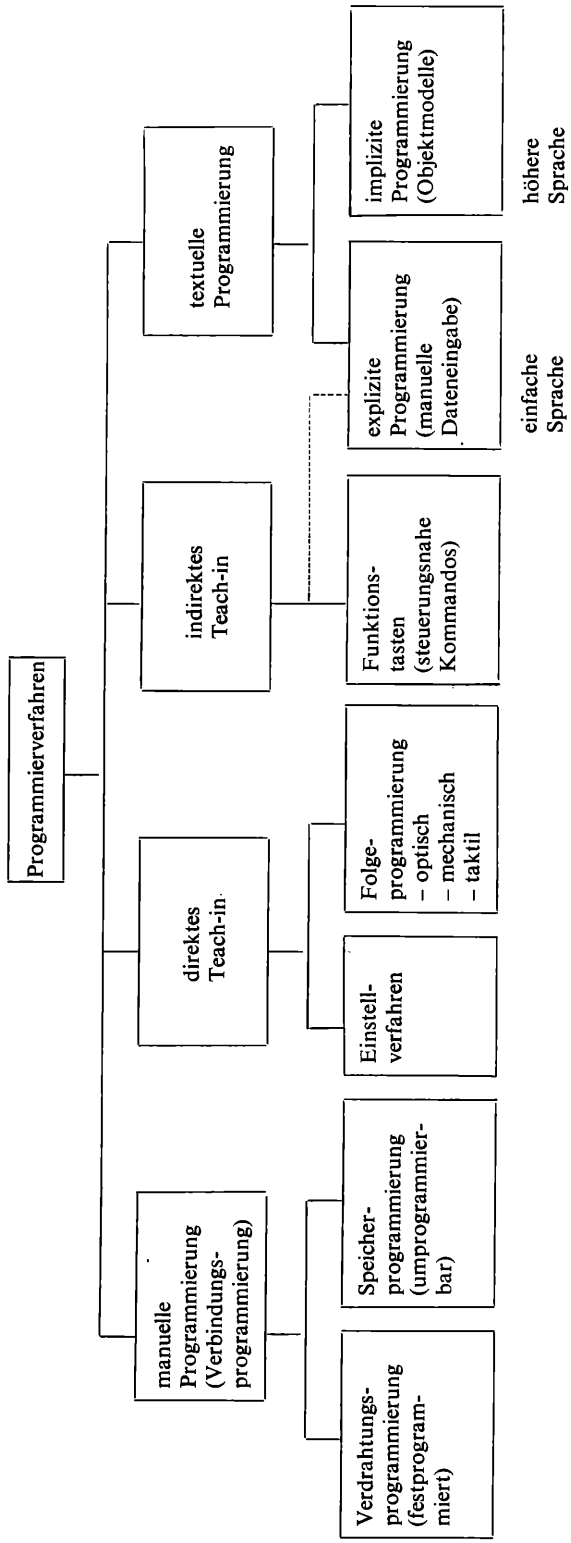


Bild 6.1. Programmierverfahren für Industrieroboter

Tafel 6.1. Programmierungsarten von Industrierobotern – Charakteristika und typische Einsatzfälle

		Programmierungsart			
		Manuell	Teach-in		Textuell
			direkt	indirekt	
Programmierung von	Bewegungen/ Positionen	Potentiometer Ziffern-wahlschalter	Vorführen mit echtem oder proportionalem Zeitbezug	Anfahren und Speichern	textuelle Eingabe am Bedienfeld oder Programmeingabe von einem externen Speichermedium
	Aktionen	Stecken von Verbindungen		textuelle Eingabe am Programmiergerät	
	Programmabläufen		textuelle Eingabe am Bedienfeld		
Typische Einsatzfälle	Beschicken	▬▬▬		▬▬▬	
	Farbspritzen		▬▬▬	▬▬▬	
	Punktschweißen			▬▬▬	
	Lichtbogenschweißen		▬▬▬	▬▬▬	▬▬▬
	Montage			▬▬▬	▬▬▬

Die manuelle Herstellung der Verbindungen hat nur sehr wenig gemein mit dem üblichen Begriff Programmierung. Allerdings kann man solche Verbindungen auch durch Umsetzen von Logikplänen realisieren und in einer Textform (Programm) beschreiben (s. Abschn. 8.4.).

6.2.2. Teach-in-Programmierung

Zur Teach-in-Programmierung werden alle Verfahren gerechnet, bei denen die Bewegungsbahn durch Bewegen oder Verfahren des IR-Grundgeräts definiert wird. Das Einstellen der gewünschten Positionen kann dabei auf verschiedene Weise erfolgen. Man unterscheidet diese Verfahren nach der Art der Bewegung des IR-Geräts in

- direktes Teach-in
- indirektes Teach-in.

Allen Teach-in-Verfahren ist gemeinsam, daß über die Positionsmeßsysteme die Stellungen der einzelnen Achsen zu bestimmten Zeitpunkten eingelesen und in einen Speicher für die spätere Bearbeitung der Programme abgelegt werden (Play-back-Verfahren, Taperecordermode). Der IR wird bei der Programmierung als Meßmaschine benutzt.

6.2.3. Direktes Teach-in

Bei diesem Verfahren wird der IR in der Programmierphase direkt bewegt. Die einfachste Art besteht darin, den Effektor bei gelösten Bremsen von Hand in die gewünschten Positionen zu bringen und die aktuellen Werte der Lagesensoren auf Anweisung (Funktionstaste) einzuspeichern. Das Verfahren wird auch *Einstellverfahren* genannt.

Der Zeitaufwand für die Programmierung einer komplexen Aufgabe kann sehr hoch werden, wenn die Zahl der einzustellenden Punkte groß ist. Das Einstellverfahren muß zur vollständigen Definition von Bahnen zusätzliche Komponenten zur Einstellung des zeitlichen Ablaufs (Geschwindigkeit oder Zeit) enthalten. Diese Notwendigkeit besteht nicht, wenn die Steuerung den Bewegungsablauf zu bestimmten Zeitpunkten selbständig abtastet und die aktuellen Bahndaten (Bewegungspunkte und durchzuführende Aktionen) in den Speicher überträgt. Ein solches Verfahren nennt man auch *Folgeprogrammierung*.

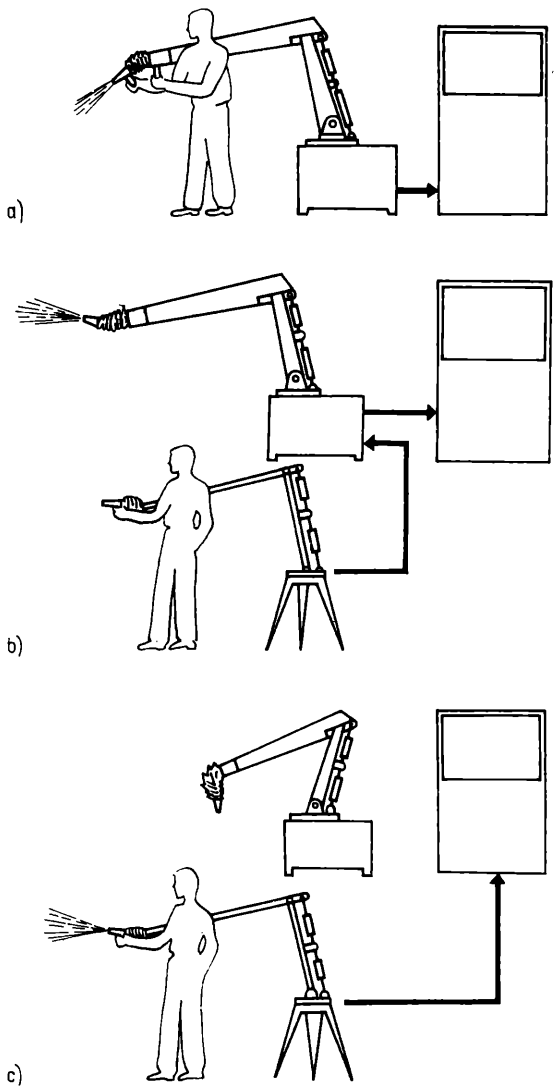


Bild 6.2
Varianten des direkten Teach-in
a) direkte Bewegung des Industrieroboters bei freigeschalteten Antrieben, Übernahme der Positionen von den IR-Meßsystemen; b) Steuerung des IR über ein Programmierhilfsgerät, Übernahme der Positionen von den IR-Meßsystemen; c) Bewegung des Hilfsgeräts allein, Übernahme der Positionen von den Meßsystemen des Hilfsgerätes

Die Folgeprogrammierung wird vorwiegend im technologischen Bereich (Farbspritzen) eingesetzt (Bild 6.2a).

Ein erfahrener Facharbeiter führt dabei den IR mit der Hand in der gewünschten kontinuierlichen Form mit der erforderlichen Geschwindigkeit.

Die Steuerung muß dabei aus dem Bahnverlauf heraus notwendige Stützstellen bilden,

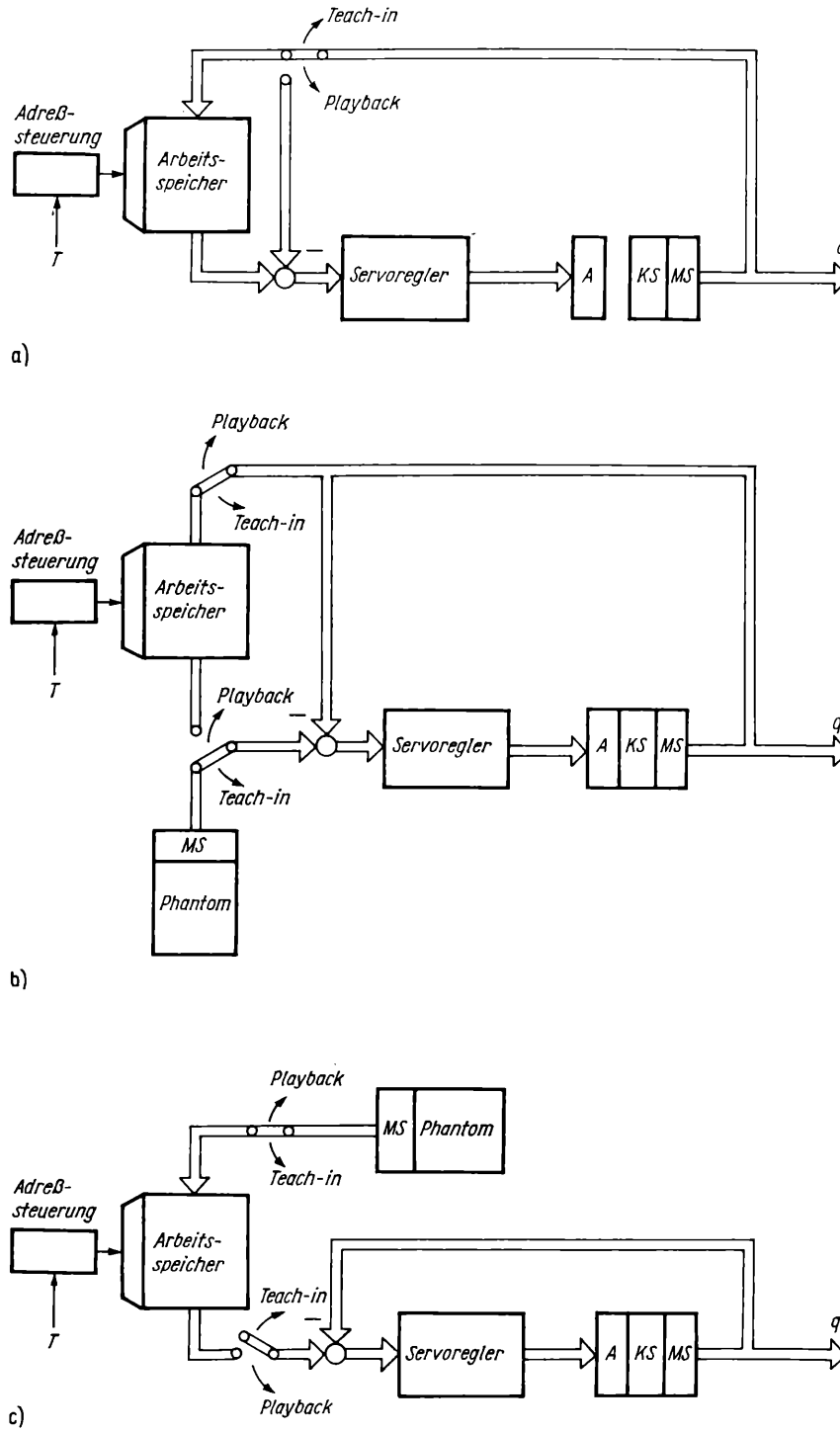


Bild 6.3. Datenflüsse bei Verfahren der direkten Teach-in-Programmierung
a) Datenfluß zu Bild 6.2a; b) Datenfluß zu Bild 6.2b; c) Datenfluß zu Bild 6.2c

zwischen denen im späteren Arbeitsablauf interpoliert werden kann. Die Anzahl der abzuspeichernden Punkte kann, in Abhängigkeit vom technologischen Programm, sehr groß sein, und es müssen geeignete Berechnungsverfahren zur Interpolation eingesetzt werden. Bei dem direkten Einstellverfahren und der beschriebenen Form der Folgeprogrammierung wird vorausgesetzt, daß sich der IR von Hand bewegen läßt. Das ist z.B. bei elektrischen Antrieben mit stark untersetzenden (selbsthemmenden) Getrieben nicht ohne weiteres möglich. Beim *Master-Slave-Verfahren* wird diese Schwierigkeit dadurch überwunden, daß ein leichter Modellroboter (Master) vom Programmierer geführt und die Bewegung vom Master auf den eigentlichen IR (Slave) übertragen wird (Bild 6.2b und c). Der Vorteil dieses Verfahrens besteht darin, daß die Programmierung für den Bediener wesentlich leichter wird und er nicht im unmittelbaren Arbeitsbereich des IR stehen muß. Die technische Ausrüstung ist allerdings wesentlich aufwendiger, da einige Komponenten doppelt bzw. als Modell aufzubauen sind. Äußerlich ist die Master-Slave-Programmierung mit der Bedienung von Teleoperatoren vergleichbar, bei denen jedoch keine Speicherung der Bahndaten erfolgt. Die Folgeprogrammierung läßt sich durch Bewegen über den eigenen Antrieb des IR realisieren, wenn der Effektor über einen optischen Sensor einer vom Programmierer geführten Lichtquelle folgt. Das gleiche kann auch mit taktilen Sensoren erreicht werden. Die Steuerung bildet dabei einen Regelkreis, für den die Sollwerte durch den Programmierer vorgegeben werden.

Bild 6.3 zeigt abschließend die den einzelnen Varianten des direkten Teach-in aus Bild 6.2 entsprechenden Verarbeitungsstrukturen.

6.2.4. Indirektes Teach-in

Beim direkten Teach-in wird der IR durch Betätigen von Funktionstasten durch seine eigenen Antriebe bewegt. Ist eine gewünschte Position eingestellt, so wird über eine weitere Funktionstaste das Einlesen der Achsstellungen von den Lagesensoren angegeben und der Punkt mit weiteren Angaben zur Bahn in dem Speicher des Steuerrechners ab-

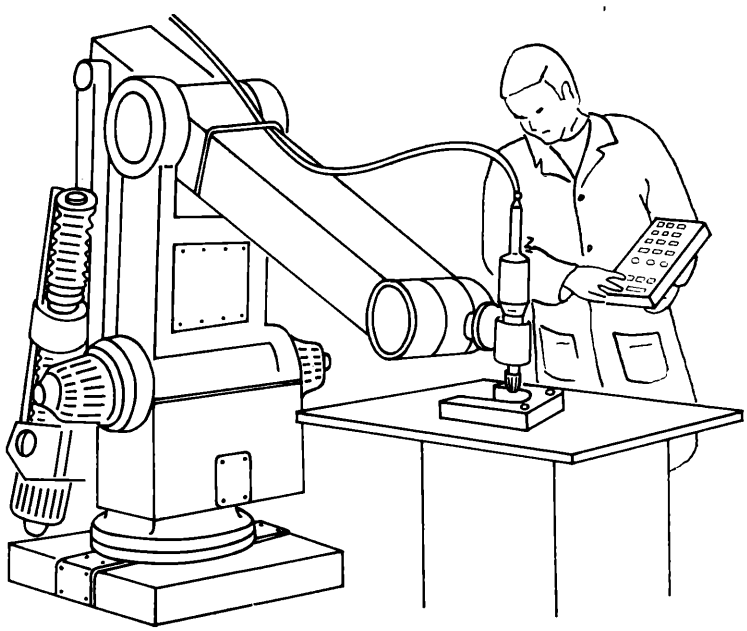


Bild 6.4. Indirekte Teach-in-Programmierung

gelegt (Bild 6.4). Zur Programmierung werden Hilfsmittel eingesetzt, um bestimmte Funktionen aufrufen zu können. Solche Programmiereinheiten sind ausgeführt als einfache Tastenfelder mit optischen Anzeigen oder bei fortgeschrittenen Systemen auch mit vollständiger alphanumerischer Tastatur und Bildschirmanzeige. Das indirekte Teach-in ist das z. Z. am häufigsten eingesetzte Programmierverfahren. Auch bei höheren Manipulatorsprachen findet man häufig zur Definition von wichtigen Punkten im Arbeitsraum Teach-in-Komponenten vor (z. B. POINTY beim Programmiersystem AL; s. Abschnitt 6.2.8.). Der Übergang zu Systemen mit textueller Programmierung ist fließend, da zu den über Funktionstasten eingegebenen Befehlen häufig weitere Angaben zur Bahnsteuerung programmiert werden müssen. Die Darstellung der Anweisungen in Textform für einfache Programmiersysteme wird im Abschnitt 6.2.6. beschrieben. Die dazu erforderlichen Teach-in-Hilfsmittel sollen im folgenden am Beispiel der Steuerung IRS 600 für den IR2 erläutert werden.

Der Bediener tastet die Arbeitsaufgabe unter Verwendung eines Teach-in-Tableaus Bild (6.5) in die Robotersteuerung ein und betätigt dabei Tasten für

- Betriebsarten
- Verfahrgeschwindigkeit
- Wegebedingungen
- Achsenpositionen
- Unterprogrammaufruf
- Programmauswahl
- Programmablauf u. ä.

Die Funktionstasten sind entsprechend ihrer Funktion beschriftet und können leicht ausgewählt werden. Im Hintergrund dieser Eingaben steht eigentlich eine Programmiersprache (z. B. NC-Sprache bei IRS 600). Das ist bei einer Programmierung über Terminal sehr leicht zu erkennen. Eine Programmierbox hat den Vorteil, daß sie wesentlich weniger Aufwand in der Realisierung erfordert als ein Terminal und beweglicher eingesetzt werden kann. Für umfangreiche Sprachsysteme, die z. B. eine werkstücksbezogene Koordinatenangabe zulassen, sind Funktionstasten mit einer starren Zuordnung nur noch bedingt geeignet.

Der Funktionsinhalt einzelner Tasten ist dann abhängig von der eingestellten Betriebsart, und vom Bediener wird eine genaue Kenntnis der zeitlichen Reihenfolge der Tastenbetätigung gefordert. Der Vorteil einer einfachen Tastenbedienung kommt nur zum Tragen, wenn das Bediengerät übersichtlich gestaltet ist und solche Funktionsumschaltungen leicht zu erkennen sind.

Die Beschreibung der Bahn erfolgt beim indirekten Teach-in weitgehend in Roboterkoordinaten, d. h., für jede Achse gibt es fest zugeordnete Tasten (bei kartesischen IR z. B. für die Achsen X, Y, Z, A, C und bei Gelenk-IR für jedes Drehgelenk eine bzw. zwei für Drehung in positiver oder negativer Richtung). Die Tasten sind vom Programmierer so lange zu betätigen, bis eine gewünschte Position erreicht ist. Der Vorteil des reinen Teach-in-Verfahrens besteht darin, daß Programme on-line erstellt, getestet und korrigiert werden können, und zwar durch den Maschinenbediener am Bearbeitungsort. Das Verfahren kann wie beim direkten Teach-in durch zusätzliche Hilfsmittel unterstützt werden. So werden z. B. Steuerknüppel eingesetzt, bei denen gleichzeitig eine Bewegung in mehreren Achsen angegeben werden kann. Durch einen Steuerknüppel lassen sich Positionen oder Bewegungsrichtungen in der x-y-Ebene einstellen. Zur Realisierung einer räumlichen Punktbeschreibung ist die z-Komponente durch einen zusätzlichen Hebel o. ä. zu realisieren.

Bildet der IR mit seinen Achsen keinen kartesischen Arbeitsraum, so ist durch die Steuerung eine Koordinatenumrechnung durchzuführen. Das Verfahren entspricht etwa dem Master-Slave-Verfahren, wobei natürlich konstruktionsbedingte Einschränkungen gelten. Der Steuerknüppel ist besonders vorteilhaft, wenn für die Bewegung im Arbeitsraum starke Restriktionen gegeben sind (z.B. Bewegung einer Schweißzange durch die Fensteröffnung einer Kfz-Karosserie). Mit dem indirekten Teach-in lassen sich sowohl einfache Punkt-zu-Punkt-Steuerungen (PTP) als auch Bahnsteuerungen (CP) realisieren. Bei der Programmierung komplexer Abläufe ist die Angabe zusätzlicher Daten zur Objekt- und Bahnbeschreibung erforderlich, und die einfache Teach-in-Programmierung wird durch Verfahren der textuellen Programmierung abgelöst.

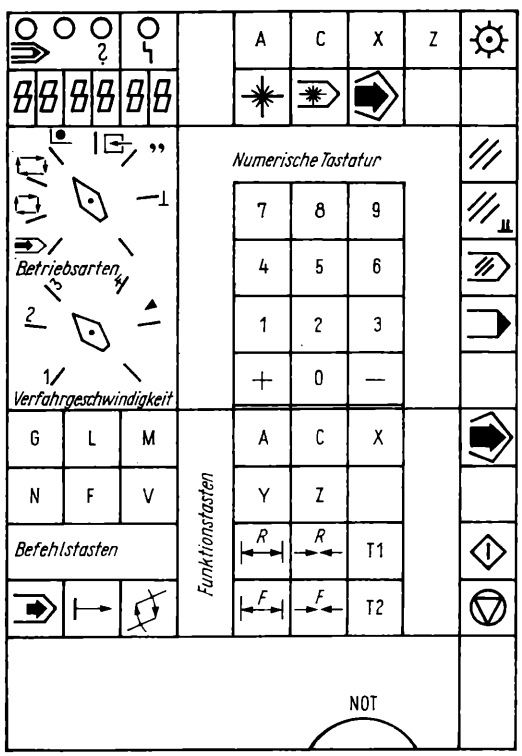


Bild 6.5. Handbedieneinheit für indirektes Teach-in (IRS 600)

Betriebsarten: Satzsuchlauf (SASU); Automatik (AUT); Schrittautomatik (SAAUT); Einrichten (EINR); Teach-in-Betrieb (TEACH); Quittierung M-Befehle (QUIM); Generierung von G-Befehlen und von Variablen
Befehlstasten: Generierung der M-Befehle (GEN); *G* G-Befehle (Zeit- und Wegebedingungen); *L* L-Befehle (UP-Nummer); *M* M-Befehle (Hilfsbefehle); *N* N-Befehle (Satznummer); *F* F-Befehle (Verfahrgeschwindigkeit); *V* V-Befehle (Variable für M-Befehle); Leserstart (LES); Stanzerstart (in Verbindung mit KOR); Ausführung eines eingegebenen; Satzes (REAL); Korrigieren (KOR)
Funktionstasten (wirksam bei EINR und TEACH): *A* Ansteuerung Achse *A*; *C* Ansteuerung Achse *C*; *X* Ansteuerung Achse *X*; *Y* Ansteuerung Achse *Y*; *Z* Ansteuerung Achse *Z*; Rohteilgreifer öffnen/schließen (GR OE, GR S); Fertigteilgreifer öffnen/schließen (GF OE, GF S); *T1*, *T2* Takteinrichtung 1/2 drehen um einen Schritt
Numerische Tastatur: + — Richtung positiv/negativ; Anzeige/Löschen bei G- und V-Befehlen; 0 bis 9 Zifferneingabe

Als Basiszyklus sind beim indirekten Teach-in immer folgende Schritte erforderlich:

- Einstellen der Verfahrgeschwindigkeit (Schalter, Potentiometer);
- Betätigen von Tasten, die eine Bewegung der zugeordneten Achsen bewirken;
- Speichertaste betätigen, wenn Zielpunkt erreicht.

Zusätzliche Angaben zur Bahn (z.B. Bewegung auf einer Geraden) werden durch gesonderte Tasteneingaben realisiert. Ein Beispiel für die IRS-600-Programmierung wird im folgenden Abschnitt in textueller Form angegeben. Die Texte entstehen durch entsprechende Tastenauswahl.

6.2.5. Textuelle Programmierung

Textuell bedeutet hier, daß der Algorithmus, der die Aufgabenstellung umsetzt, in Form eines lesbaren Textes erstellt und vom Programmiersystem in eine für die IR-Steuerung geeignete Form umgesetzt wird. Der Text ist eine symbolische Beschreibung von Operationen und Daten in Form von zulässigen Zeichenfolgen einer Programmiersprache.

Die textuelle Formulierung einer Aufgabe bietet im Vergleich zur reinen Teach-in-Programmierung eine Reihe von *Vorteilen* [6.10]:

- Das Programm ist für den Programmierer und andere Anwender lesbar (mitteilbar).
- Das Programm läßt sich leicht ändern (editierbar).
- Daten sind als Variablen darstellbar, deren Wert erst zur Ablaufzeit ermittelt und eingetragen wird (z.B. Sensordaten).
- Das Programm kann in lesbarer Form gespeichert werden (Dokumentation).
- Das Programm kann off-line (ohne IR) erstellt und teilweise kontrolliert und getestet werden.

Als *Nachteile* der textuellen Programmierung sind zu nennen:

- Das Programmiersystem ist komplexer und aufwendiger in der Realisierung (Hardware und Software).
- Die Bewegungspunkte lassen sich nicht vollständig off-line beschreiben; es ist meistens zusätzlich eine Teach-in-Komponente erforderlich, die on-line in einem Testlauf die fehlenden Angaben ergänzt.
- Die Programmiersprache erfordert i.allg. eine höhere Qualifikation des Programmierers als eine einfache Tastenbedienung. Die erforderliche Qualifikation ist natürlich von der Anpassung der Sprache an die Problemstellung abhängig (Fachsprache).

Die textuelle Programmierung gewinnt gegenüber der reinen Teach-in-Programmierung ständig an Bedeutung. Als eine Variante der textuellen Programmierung ist die *akustische* Programmierung anzusehen, bei der ein Text nicht als symbolische Zeichenfolge über Tastatur oder Datenträger, sondern über ein Sprachanalysesystem eingegeben wird. Die Komplexität der Sprache wird dabei vorwiegend durch die Leistungsfähigkeit des Sprachanalysesystems bestimmt. Ein Vorteil der akustischen Programmierung liegt darin, daß die Eingabe ohne mechanische Hilfsmittel erfolgen kann und der Programmierer bei der Festlegung des Arbeitsablaufs die Hände frei hat für andere Aufgaben (z.B. Werkstückzuführung), die später durch andere technologische Einrichtungen realisiert werden.

6.2.6. Daten und Befehle im Roboterprogramm

Ein Programm besteht formal aus zwei Bestandteilen:

- Objektbeschreibungen in Form von Datenstrukturen,
- Befehlen, die mit den Datenstrukturen operieren.

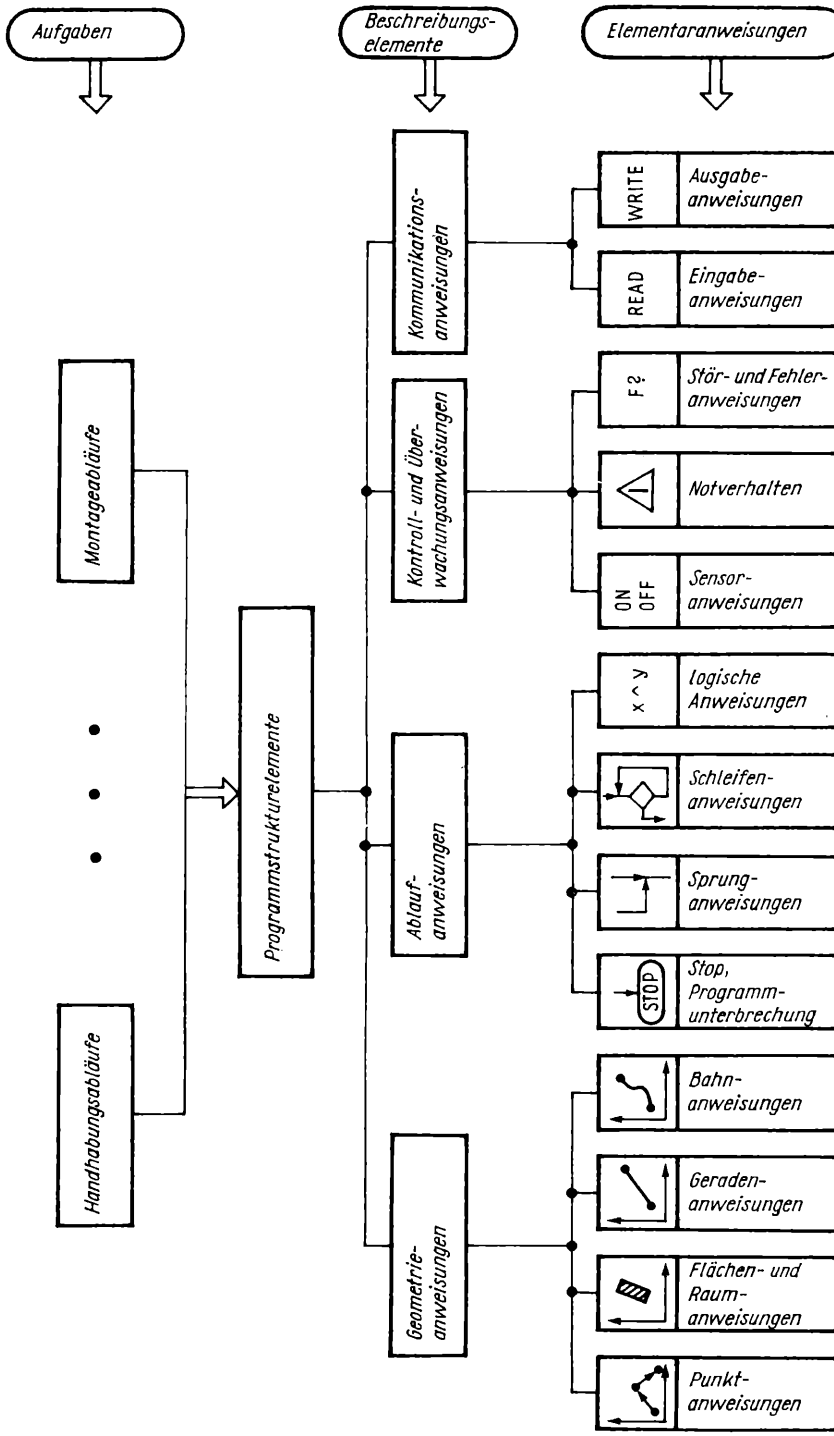


Bild 6.6. Programmstrukturelemente für IR-Programme (nach [6.11])

Programme in einer IR-Programmiersprache unterscheiden sich von Programmen in einer normalen Rechnerprogrammiersprache dadurch, daß spezifische Datenstrukturen (z. B. Bewegungspunktbeschreibungen) und spezielle Befehle enthalten sind. Programmstrukturelemente von Industrieroboterspracheln sind im Bild 6.6 zusammengefaßt. Diese Sprachelemente dienen zur Beschreibung von

- Trajektorien (Positionen, zeitlicher Ablauf),
- Werkstückgeometrie und Relationen zwischen Objekten,
- Sensordaten,
- Programmstrukturen,
- Steuerungsabläufen und möglicherweise
- arithmetischen Operationen.

Allgemein lassen sich die Daten (Objekte) als Elemente eines Umweltmodells auffassen, das entweder explizit im Programm aufgestellt (explizite Programmierung) oder aus einer Datensammlung generiert wird (implizite Programmierung).

Beim Entwurf einer IR-Sprache kann man prinzipiell zwei Wege gehen. Entweder wird eine vorhandene Rechnerprogrammiersprache um roboterspezifische Konstruktionen ergänzt, oder es wird eine gesonderte Sprache entwickelt. So existieren heute einerseits Sprachen, die wie PASCAL oder BASIC aufgebaut sind, und andererseits spezielle IR-Sprachen, die auch Erweiterungen von NC-Sprachen darstellen können.

Auf einige Sprachrealisierungen wird in den folgenden Abschnitten eingegangen. Eine vollständige Übersicht ist nur noch schwer zu erreichen, da die Zahl der Sprachen bereits recht groß ist [6.27]. Es ist zu erwarten, daß sich auch hier internationale Standards herausbilden werden, wie das im Bereich der NC-Sprachen bereits geschehen ist.

Beim Entwurf einer Sprache ist festzulegen, welche Objekte (Datenstrukturen) durch welche Operationen bearbeitet werden können. Diese Festlegungen sind dann in Form der Syntax der Sprache für die Programmierung verbindlich. Abweichungen werden durch das Programmiersystem erkannt und als Fehler angezeigt.

6.2.7. Einfache Manipulatorsprachen

Ein Programm in einer einfachen Manipulatorsprache beschreibt explizit die Bewegung des Effektors auf einer durch Bewegungspunkte im Arbeitsraum des IR beschriebenen Trajektorie und die zugehörigen Aktionen. Dabei müssen alle Komponenten der Steuerung und des Manipulators durch entsprechende Sprachelemente angesprochen werden können. Diese Sprachebene realisiert damit eine virtuelle (programmierbare) Maschine, deren Befehle in Echtzeit on-line den Roboter steuern. Das entspricht der Assemblerprogrammierung eines Rechners, bei der alle Rechnerfunktionen durch Maschinenbefehle realisiert werden.

Zur Formulierung der Arbeitsaufgabe in einer einfachen Manipulatorsprache wird der Programmtext über einen Editor in den Speicher eingetragen und nach einfachen Kriterien überprüft. Der Programmierer hat die Möglichkeit, den Text beliebig zu ändern und auf einem Datenträger (z. B. Magnetband oder Floppydisc) auszugeben. Die Umsetzung des Textes in eine für die Steuerung geeignete Form ist von der Komplexität der Sprache und dem zugehörigen Programmiersystem abhängig. Im einfachsten Fall wird die Eingabeform direkt vom Editor syntaktisch überprüft und in eine kompakte interne Darstellung umgewandelt, die dann vom Interpreter ausgewertet wird. Die Definition der Bewegungspunkte kann dabei während der Programmeingabe über Teach-in erfolgen

oder in einem separaten Testlauf, bei dem zunächst symbolisch angegebene Bahnpunkte interaktiv angefordert oder auch durch Berechnung bereitgestellt werden.

Je nach Ausführung der Steuerung erfolgt die Programmierung der Bewegungspunkte in robotereigenen und/oder in Umweltkoordinaten.

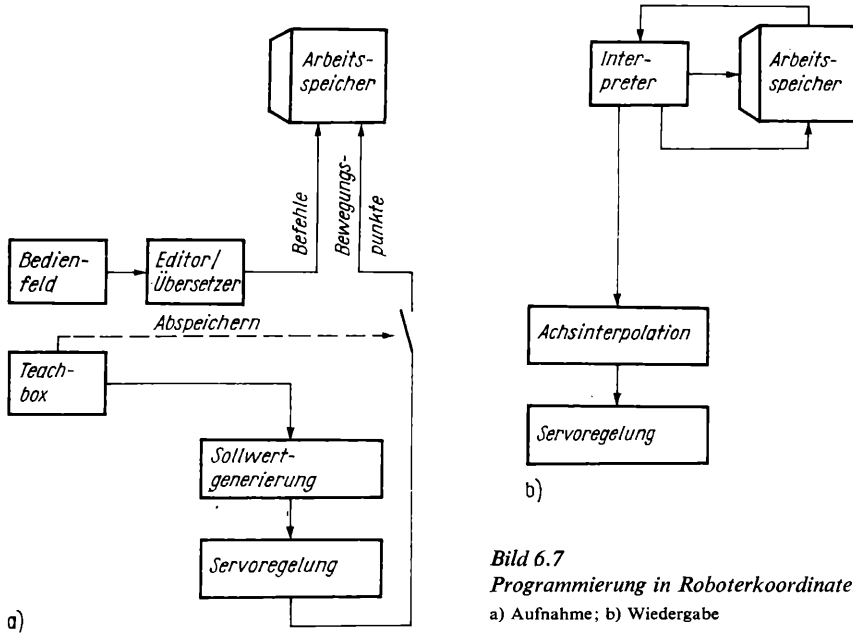


Bild 6.7
Programmierung in Roboterkoordinaten
a) Aufnahme; b) Wiedergabe

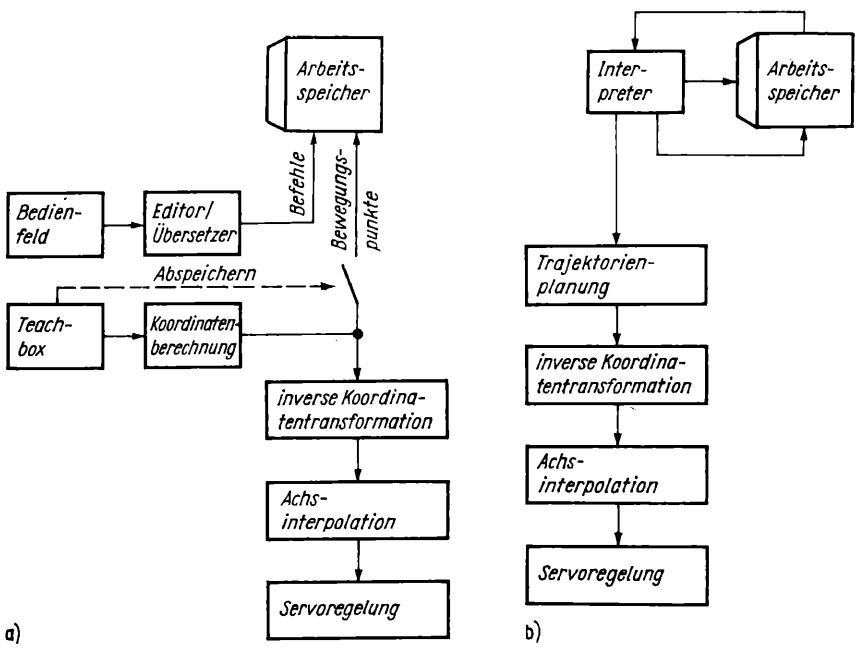


Bild 6.8. Programmierung in Umweltkoordinaten
a) Aufnahme; b) Wiedergabe

Tafel 6.2. IR-Sprache TIPS II [6.12]

TIPS II

Pseudo-Kommandos		Programm-Kommandos	
Geschwindigkeitsangabe		Geometrieanweisungen	
VA	Automatikbetrieb	GP	Grobpositionierung
VK	Korrektur im Befehl	FP	Feinpositionierung
VT	Testbetrieb	BG	berechnete Grobposition
VM	Handbetrieb	BF	berechnete Feinposition
Kontrollfunktionen		LP	Linearpositionierung
SN	Schnappschuß	CP	Bahnsteuerung
LD	Laden Programm von ROM	Ablaufanweisungen	
UM	Umschalten Programmbereich	LB	Marke Setzen
SU	Substitution von Byte	JU	Sprung
Betriebsarten/Ablaufsteuerung		JC	Sprung bedingt
TI	Teach-in im RAM	CA	Unterprogrammaufruf (UP)
EX	Beenden Teach-in	CC	UP-Aufruf bedingt
ST	Einzelschritt-Modus	RE	UP-Rücksprung
LI	Wiederholter Einzelschritt	RC	UP-Rücksprung bedingt
CTRL/C	Halt (STOP)	DO	Programmschleife von/bis
CTRL/X	Abbruch (ABORT)	BR	Abbruch Programmschleife
		CL	Abarbeitung Kommandofolge
Editor-Kommandos		SF	Merkbit Setzen/Rücksetzen
		BS	Byte-Substitution
B	Zeiger auf Beginn	SE	Suchfunktion (Monitor)
Z	Zeiger auf Ende	SA	Abbruch Suchfunktion
±n L	Zeiger n Zeilen vorwärts/rückwärts	SP	Start Nebenprogramm
±n T	Terminalausgabe von RAM-Zeilen	WP	Warten auf Nebenprogramm
	CTRL/S = STOP	WT	Warten von Zeitfakten (0,1 s)
	CTRL/Q = CONTINNE	Kontroll-/Überwachungsanweisungen	
	CTRL/C = CANCEL	WB	Warten auf Binäreingabe
±n K	Streichen von RAM-Zeilen	BO	Binärausgabe
		GA	Greifer auf
		GZ	Greifer zu
⟨ESC⟩	Aufruf des Editors in Betriebsart TI	Sonstige Anweisungen	
⟨ESC⟩	⟨ESC⟩ Abschluß eines Editorkommandos	KO	Kommentarzeile

Bei der *Programmierung in Roboterkoordinaten* bleibt das Programm weitgehend vom konkreten Robotertyp abhängig. In der Programmierphase werden die Bewegungspunkte über die Teachbox angesteuert und in roboterspezifischen Koordinaten abgespeichert (Bild 6.7a). Geschwindigkeiten, Aktions- und Programmorganisationsbefehle werden über das Bedienfeld eingegeben. Während der Programmausführung übernimmt der Interpreter die Bewegungspunkte mit den zusätzlichen Angaben aus dem Speicher. Die Trajektorienplanung ist auf eine einfache Interpolation in Gelenkkoordinaten beschränkt. Sie stellt die Sollwerte für die Servoregelung bereit (Bild 6.7b).

Bei der *Programmierung in Umweltkoordinaten* werden die Position und Orientierung des Effektors in natürlichen, meist kartesischen Koordinaten abgespeichert (Bild 6.8a). Damit ist auch die Anwendung des Framekonzepts (s. Abschn. 4.6.7.) für den Anschluß äußerer Sensoren gewährleistet. Beim Anfahren der Bewegungspunkte und bei der Programmarbeitung macht sich eine inverse Koordinatentransformation (von Umweltkoordinaten in roboterspezifische Koordinaten) erforderlich. Die Trajektorienplanung

Tafel 6.3. Sprache der IRS 600 [6.14]

IRS 600

Pseudo-Kommandos (Tasten)

Geschwindigkeitsangabe
Maximalwert generiert
Wahlschalter zur Reduzierung
F 1 ... 4

Kontrollfunktionen
Anzeigetasten
LES Programmeingabe LS
LES/KOR Programmausgabe LS
KOR Korrigieren
REAL Satzausführung

Betriebsarten
SASU Satzsuchlauf
AUT Automatikbetrieb
SAAUT Schrittautomatik
TEACH Teach-in
EINR Einrichten
GEN Generieren von G-Befehlen und Variablen V der M-Befehle
QUIM Quittierung M-Befehle
AUP Programmwahl
UEB Übernahme
START Programmstart
STOP Programmstop
NOT Havarieschalter

Editor-Kommandos

LOE P Programm Löschen
LOE S Satz löschen
LOE Z Zeichen löschen

Programm-Kommandos

Geometrieanweisungen
A Positionierung A-Achse
C Positionierung C-Achse
X Positionierung X-Achse
Y Positionierung Y-Achse
Z Positionierung Z-Achse
G 20 ... G79 Festwerte

Ablaufanweisungen
G Zeit- und Wegebedingungen
G0 ... G3 Warten Zeitglied (100 ms)
G4 ... G7 Koinzidenzbereiche
G8 ... G11 Generierung von Relativwegen
G12 ... G19 Addition/Subtraktion von Relativwegen
G20 ... G39 Festpos. Achse Z
G40 ... G59 Festpos. Achse X
G60 ... G79 Festpos. Achse C
G80 ... G87 Festwerte von WZM 1 ... 7
G90 ... G94 Festwerte von Paletten 1 ... 4
G98 ... G99 Zählrichtung (M56 ... M59)
L01 ... L98 Unterprogramme
M Hilfsbefehle
MØ Programmstop
M2 Programmende
M3 ... M23 Warten auf Binärsignal
M24 ... M39 UP-Rücksprung bedingt
M40 ... M47 Zyklusende bedingt
M94 UP-Ende
M95 Zyklusbeginn
N01 ... N99 Satznummer

Variablenoperationen

M56 ... M59 Zwischenspeicher 1 ... 4 ändern um 1
M60 ... M91 Setzen Variablen 53 ... 84
M92 Löschen Variablen 53 ... 84
M93 Setzen Variablen 53 ... 84
M96 ... M99 Löschen Zwischenspeicher 1 ... 4
M100 ... M131 Löschen Variablen 53 ... 84

Variablenvereinbarung für M-Befehle

V1 ... V11 Schließer
V12 ... V16 Schließer und Öffner
V17 ... V24 UP-Ende
V25 ... V32 Zyklus-Ende
V33 ... V40 zu puffernde Impulse
V41 ... V44 Zwischenspeicherbedingung
V45 ... V52 Puffer zu V33 ... V40
V53 Rohteilgreifer
V54 Fertigteilgreifer
V55 Palettentakteinrichtung 1
V56 Palettentakteinrichtung 2

V84 Ausgangssignale
V85 ... V88 Zwischenspeicher 1 ... 4

umfaßt jetzt die Generierung von Raumkurven aus wenigen Vorgaben und umfassende Modifikationen von Bewegungsdaten. Dadurch können Bewegungsprogramme wesentlich kompakter gespeichert und einfacher geändert werden.

Als Beispiel für eine einfache Programmiersprache mit integrierter Teach-in-Komponente für das Roboterkoordinatensystem soll die Sprache TIPS II zur Steuerung der Gelenkroboter ZIM 60 und ZIM 10 dienen. In Tafel 6.2 ist die Syntax von TIPS II zusammengestellt [6.12] [6.13]. Die Eingabe von Roboterkoordinaten erfolgt während der Programmerstellung im indirekten Teach-in (Befehle GP, FP, LP, CP).

Eine weitere einfache Programmiersprache ist die von der NC-Programmierung abgeleitete Form der IRS 600 für den Beschickungsroboter IR 2. Als Vorteil dieser Sprache ist zu nennen, daß sie die Erfahrungen der NC-Programmierung direkt übernimmt und so eine einfache Programmierung von Robotern an NC-Maschinen ermöglicht. Nachteilig ist, daß sie nur wenige Elemente zur Programmablaufsteuerung enthält und nicht sehr anschaulich ist. Tafel 6.3 enthält eine Syntaxbeschreibung der IRS-600-Programmiersprache [6.14]. Die Roboterkoordinaten werden explizit oder durch Anfahren und Abspeichern bereitgestellt.

Das folgende Beispiel aus der Bedienungsanleitung [6.14] soll die Arbeitsweise mit einer IR-Sprache auf NC-Basis deutlich machen.

Beispiel IRS-600-Programmierung

Aufgabenstellung für IR2 (3 NC-, 2 Nockenachsen).

Ein Teil soll von der Position 1 ($x = 10000, z = 20000, c = 12000$) mit dem Rohteilgreifer geholt, auf Position 2 ($x = 30000, z = 50000, c = 500$) abgelegt, mit dem Fertigteilgreifer wieder aufgenommen und zurück zur Position 1 gebracht werden. Dieser Vorgang soll zehnmal wiederholt werden. Die Ausgangsposition ist Position 3 ($x = 1000, z = 1000, c = 100$). Geschwenkt werden soll mit $x = 5000$. Zur Angabe der Festpositionen werden G-Befehle generiert und für Hilfsfunktionen M-Befehle genutzt.

G- und M-Befehle

G0	=	10	Wartezeit 1 s
G12	=	5000	5 cm Abhebemaß in Z
G17	=	4800	4,8 cm Absenkmaß in Z für Ablage
G21	=	20000	20 cm Position 1 in Z
G22	=	50000	50 cm Position 2 in Z
G23	=	1000	1 cm Position 3 in Z
G41	=	10000	10 cm Position 1 in X
G42	=	30000	30 cm Position 2 in X
G43	=	1000	1 cm Position 3 in X
G44	=	5000	5 cm Schwenkposition in X
G61	=	12000	Position 1 in C
G62	=	500	Position 2 in C
G63	=	100	Position 3 in C
M40	=	V25 0086	Zyklusende bei erreichtem Grenzwert
M56	=	V41 0004	Zähler erhöhen, falls Greifer offen
M60	=	V53 0052	Rohteilgreifer öffnen
M100	=		schließen
M61	=	V54 0152	Fertigteilgreifer öffnen
M101	=		schließen

Zähler (Zwischenspeicher 1)

V85 = 0010 Anfangswert 00, 10 Zyklen

Programm

% L1		Unterprogramm Nr. 1
N01	G23 G43 G63	Position 3
	M96	Löschen Zähler 1
	M60 M61	Greifer öffnen
	Y2	*Y*-Achse Stellung 2
N02	M95	Zyklusbeginn
N03	G21 G12 A3	*Z*-Position 1, oberhalb Teil
N04	G41 G61	*X*, *C*-Position 1
N05	G21	*Z*-Position 1
N06	M100 G0	Rohteilgreifer schließen, warten
N07	G12	Abheben
N08	G44	Schwenkposition für *X*
N09	G62	Schwenken in *C*-Position 2
	G22 G12	*Z*-Position 2, oberhalb Ablage
N10	G42	*X*-Position 2
N11	G17	Absenken
N12	M60	Rohteilgreifer öffnen
N13	G12	Abheben
N14	G44	*X* Zurückfahren
N15	A1	*A* Schwenken
N16	G42	*X* wieder vor (Position 2)
N17	G22	*Z*-Position 2
N18	M101 G0	Fertigteilgreifer schließen, warten
N18	G12	Abheben
N20	G44	Schwenkposition für *X*
N21	G61	Schwenken in *C*-Position 1
	G21 G12	*Z*-Position 1, oberhalb Ablage
N22	G41	*X*-Position 1
N23	G17	Absenken
N24	M61	Fertigteilgreifer öffnen
N25	G12	Abheben
N26	G44	*X* Zurückfahren
N27	M56	Zähler erhöhen
N28	M40 M2	Zyklus beenden, wenn Grenzwert
N29	M94	Unterprogrammende

An diesem Beispiel wird bereits deutlich, daß die Programmierung der Aufgabe sehr gut vorbereitet werden muß und ein Verständnis der Befehle wegen fehlender Mnemonik nur mit Tabellen möglich ist.

Die NC-Sprachen sind in einigen Anwendungsfällen stärker auf die IR-Steuerung zugeschnitten und weiterentwickelt worden. Etwa die gleiche Leistungsfähigkeit wie die IRS-600-Sprache bietet die Sprache RPL [6.15]. Der sehr formale Satzaufbau ist durch eine mehr problemorientierte, symbolische Darstellung der Anweisungen ersetzt (Tafel 6.4)

Als Beispiel einer Sprache zur Programmierung in Umweltkoordinaten wird im folgenden die Sprache VAL für den IR PUMA (Unimation) dargestellt [6.7] (Tafel 6.5). Die Programmierung der Bewegungspunkte erfolgt durch die Angabe von Frames (s. Abschnitt 4.6.7.) und Transformationen als Beziehungen zwischen Frames im Teach-in-Verfahren. Die Programmform ist an die Rechnerprogrammiersprache BASIC angelehnt.

Tafel 6.4
IR-Sprache RPL [6.15]

RPL	
Programm-Kommandos	
Geometrieanweisungen	
X	x-Translation
Z	z-Translation
C	c-Rotation
D	D-Rotation (Hand)
E	E-Rotation (Hand)
S	Überschleifparameter
ABS	Absolute Koordinaten
REL	Relative Koordinaten
Ablaufanweisungen	
WAIT	Warten von Zeittakten (0,1 s)
STOP	Programmstop
GOTO	Sprung
DO	Schleifenbeginn
CONTINUE	Schleifenende
CALL	UP-Aufruf
ENTRY	UP-Einsprung
RETURN	UP-Rücksprung
IF...	
WAIT	Warten bedingt
STOP	Stop bedingt
GOTO	Sprung bedingt
CALL	UP-Aufruf bedingt
RETURN	UP-Rücksprung bedingt
T1 ON	Ausgabesignal bedingt
Kontroll- und Überwachungsanweisungen	
F	Verfahrensgeschwindigkeit

Die Verarbeitung erfolgt durch einen Interpreter. Das Programm bleibt dadurch stets im Quelltext lesbar, und Programmkorrekturen lassen sich sehr einfach ausführen. Zur Synchronisation mit dem technologischen Prozeß sind einfache binäre Eingabe- und Ausgabebefehle vorhanden. Der Anschluß von Sensoren, die den Zahlenwert einer Meßgröße liefern, ist nicht vorgesehen. Eine Spracherweiterung für das visuelle Erkennungssystem VS 100 (UNIVISION) gestattet die Verwendung von Positionsvariablen, deren Werte erst zur Ausführungszeit festgelegt werden [6.16]. Durch die Anwendung von relativen und zusammengesetzten Koordinatenvorgaben lassen sich sehr einfach Unterprogrammbibliotheken aufbauen, die zur Generierung umfangreicher Programme verwendet werden können.

Das folgende Beispiel gibt ein einfaches Programm zum Entladen einer Palette an. Dabei wird ein Unterprogramm zur Bestimmung der Palettenpositionen aufgerufen.

Tafel 6.5. IR-Sprache VAL [6.7]

VAL

Monitor-Kommandos		Programm-Kommandos	
Bewegungspunktdefinition		Geometrieanweisungen	
POINT	Zuweisung	ALIGN	
DPOINT	Löschen eines Punktes	APPRO (S)	definierte Annäherung
HERE	aktuelle Position	DEPART (S)	definierte Entfernung
WHERE	aktueller Zustand	DRAW	Positionierung relativ
TEACH	Zuweisung einer Folge im Teach-Modus	DRIVE	Ansteuerung einer Achse
BASE	Basissystem	MOVE (S)(T)	Positionierung
TOOL	Werkzeugtransformation	NEST	Ruheposition
Geschwindigkeitsangabe		READY	Bereitschaftsposition
SPEED	Monitorgeschwindigkeit	ENABLE CP	kontinuierliche Bahn
Betriebsarten		FRAME	Bezugssystem
EXECUTE	Programmabarbeitung	HERE	aktuelle Transformation
ABORT	Abbruch	INVERSE	inverse Transformation
NEXT	Einzelschritt	SET	Zuweisung
CALIBRATE	Kalibrierung	SHIFT	Verschiebung
DO	Befehlsausführung	TOOL	Werkzeugtransformation
STATUS	Zustandsangabe	COARSE	Grobtoleranz
		FINE	Feintoleranz
		BASE	Basisbezugssystem
EDIT	Start Editor	Ablaufanweisungen	
C	Programmwahl	GOTO	Sprung
D	Löschen von Zeilen	GOSUB	UP-Aufruf
E	Ende Editor	RETURN	UP-Rücksprung
I	Einfügen	IF THEN	bedingter Sprung
L/P	Positionieren vor/zurück	STOP/HALT	Programmende
R	Ersetzen	Kontroll- und Überwachungsanweisungen	
S	Positionieren auf Schritt	IFSIG THEN	bedingte Reaktion
T (S)	Teach in Gelenkkoordinaten (analog TEACH)	REACT (I)	Start Bedingungsmonitor
		IGNORE	Ende Bedingungsmonitor
Bibliotheksverwaltung		WAIT	Warten auf Eingabesignal
DIRECTORY	Inhaltsverzeichnis	SIGNAL	Ausgabesignal
LISTL	Auflistung von Punkten	SPEED	Verfahrensgeschwindigkeit
LISTP	Auflistung von Programmen	OPEN (I)	Öffnen des Greifers
FORMAT	Formatieren Floppy	CLOSE (I)	Schließen des Greifers
		GRASP	Greifen
		SETI	Integerzuweisung
		Kommunikationsanweisungen	
		TYPE (I)	Ausgabe von Zahl/Text
		PAUSE	Ausgabe und Eingabe
		REMARK	Kommentar

Beispiel

VAL-Programm zum Entladen einer Palette

```
BASE    0.00, 0.00, 0.00, 30.00
SETI    MAX. REIHE = 6
SETI    MAX. SPALTE = 12
SETI    REIHE = 1
SETI    SPALTE = 1
SET     POSIT = ANFANG
```

```
20 APPRO POSIT = 50.00
MOVES    POSIT
CLOSET   0.00
MOVE     ...
GOSUB    PALETTE
GOTO     20
RETURN 0
```

Unterprogramm PALETTE

```
REMARK   NEUE PALETTENPOSITION
SETI     SPALTE = SPALTE + 1
IF       SPALTE GT MAX. SPALTE THEN 10
SHIFT    POSIT BY 30.00, 0.00, 0.00
RETURN 0
10 SETI REIHE = REIHE + 1
IF REIHE GT MAX. REIHE THEN 20
SHIFT POSIT BY - 300.00, 50.00, 0.00
SETI SPALTE = 1
RETURN 0
20 RETURN 1
```

Die Position ANFANG ist im Teach-in-Verfahren bei Arbeitsbeginn zu definieren.

Die Programmiersprache VAL ist Teil eines vollständigen Steuerungssystems, das über einen Monitor auch die interaktive Abarbeitung von Programmbefehlen und das Programmtesten ermöglicht. Einfache Manipulatorsprachen lassen sich weitgehend interpretativ abarbeiten. Dabei erweist sich die Übersetzung der Eingabesprache in die Form einer virtuellen IR-Maschinensprache als vorteilhaft. Dieser Zwischenkode läßt sich so aufbauen, daß über einen Interpreter alle Grundfunktionen der Steuerung aufgerufen werden können.

6.2.8. Höhere Programmiersprachen für Industrieroboter

Wie bei der Rechnerprogrammierung, so ist man auch für Industrieroboter bemüht, höhere Programmiersprachen zu entwickeln, um die Arbeitsabläufe im technologischen Prozeß kürzer und mehr problemorientiert formulieren zu können. Während die einfachen Sprachen sich vorwiegend auf den Endeffektor beziehen, sind in höheren Manipulatorsprachen auch Elemente zur Beschreibung von Objekten und darauf zugeschnittenen Operationen vorhanden. Die erste und bekannteste Sprache höheren Niveaus ist die Sprache AL [6.5] [6.21], die auch Grundlage für viele ähnliche Forschungsarbeiten geworden ist [6.17] [6.18] [6.19] [6.20].

Die Syntax von AL zeigt eine ALGOL-ähnliche Struktur. AL-Programme sind blockstrukturiert. Alle im Programm verwendeten Variablen müssen definiert werden. Zur Dokumentation und Erläuterung können Bemerkungen in den Text eingefügt werden.

Die grundlegenden Datentypen sind:

SCALAR Gleitkommazahlen (REAL)

VECTOR Dreitupel zur Darstellung von Translationen, Geschwindigkeiten, Positionen u.ä.

ROT 3×3-Matrix zur Orientierungsangabe

FRAME lokales Koordinatensystem (Position und Orientierung relativ zum Basissystem WORLD)
TRANS Transformationen von FRAMES und VECTOR (Aufbau wie FRAME).

Alle algebraischen Datentypen lassen sich zu mehrdimensionalen Feldern (ARRAY) zusammenfassen. Einzelnen Variablen können physikalische Dimensionen zugeordnet werden (TIME, DISTANCE, ANGLE, FORCE, TORQUE, VELOCITY u.ä.). Neue Dimensionen werden durch Ausdrücke berechnet, z.B. VELOCITY VECTOR V; V ← xhat inches/sec.

Durch Angabe eines AFFIX-Statements können Relationen zwischen einzelnen FRAMES formuliert werden. Das hat zur Folge, daß z.B. durch AFFIX BOX TO ARM RIGIDLY ein fester Zusammenhang zwischen BOX und ARM entsteht und eine Bewegung von BOX automatisch eine Bewegung von ARM bedeutet. Die Verbindung kann auch wieder gelöst werden durch UNFIX, wenn BOX z.B. abgesetzt wird.

Zur Trajektorienbeschreibung wird das MOVE-Statement benutzt, das durch die Angabe diverser Zusätze eine große Variabilität zuläßt.

Die Bewegungspunkte müssen mit Hilfe einer interaktiven Teach-Komponente (POINTY) definiert werden. Die Grundform von MOVE bewegt ein FRAME zu einem Zielpunkt:

MOVE frame T0 ziel modifikation

Als Modifikation sind folgende Angaben zulässig:

VIA f1, f2, ...	Stützpunkte
VIA f WHERE VELOCITY = v DURATION = u	Stützpunkt mit Geschwindigkeits- und Zeitangabe
WITH DEPARTURE = exp	definiertes Abfahren
WITH APPROACH = exp	definiertes Annähern
WITH FORCE = Scalar	definierte Kraft
WITH TORQUE = Scalar	definiertes Moment
WITH DURATION = Scalar	Verfahrzeit

Durch Bedingungsmonitore lassen sich Trajektorien überwachen. Die Überwachung betrifft Kräfte, Momente, Zeitdauer und parallele Abläufe und läßt sich definiert an- und abschalten (ENABLE, DISABLE). Zur Steuerung des Effektors sind Elemente, die einen Greifer betreffen (OPEN, CLOSE, CENTER) und ein allgemeines Sprachelement für beliebige Geräte definiert (OPERATE). Die Programmablaufsteuerung wird durch traditionelle Kontrollstrukturen realisiert (IF, FOR, WHILE, UNTIL, CASE). Parallele Abläufe (Coroutinen), die durch COBEGIN und COEND gekennzeichnet sind, lassen sich durch SIGNAL und WAIT synchronisieren. Weitere Sprachelemente dienen der Programmtestung und der Operatorkommunikation (PAUSE, PRINT, NOTE, DUMP).

Neben Möglichkeiten der einfachen Bewegungsprogrammierung, die auch in einfachen Manipulatorsprachen vorhanden sind, zeichnen sich höhere Manipulatorsprachen durch Elemente zur strukturierten Programmierung aus. Die Zuordnung der höheren Manipulatorsprache zum Objektniveau des Programmiersystems ist darin begründet, daß z.B. durch AFFIX und UNFIX Objektrelationen ausgedrückt werden können und durch die Zuordnung von FRAMES zu Objekten und die Wahl eines geeigneten Objektnamens die Programmierung in bezug auf Objekte durchgeführt werden kann.

Das folgende Beispiel löst die Aufgabenstellung, fünf nebeneinanderliegende Quader zu einem Stapel zusammenzubauen.

AL-Beispiel

```
BEGIN „STAPELN“
FRAME POS, TURM, UEBERTURM
POS ← ... (START BEI 1. BLOCK)
MOVE BARM TO PARK
  WITH DURATION = 5*SECONDS,
TURM ← FRAME (ROT(ZHAT, 180*DEGREES),
  VECTOR (0, 20, 0)*CM);
FOR COUNT ← 1 STEP 1 UNTIL 5 DO
  BEGIN „FOR“
OPEN HAND TO 10*CM
MOVE BARM TO POS WITH APPROACH = 10*CM;
CENTER BARM
MOVE BARM TO TURM VIA UEBERTURM
  WITH DEPARTURE = 5*CM
  WITH APPROACH = 20*CM
  ON FORCE (ZHAT) > 10*GM DO STOP,
POS ← POS + VECTOR (30, 0, 0)*CM,
END „FOR“
END „STAPELN“
```

Bild 6.9 gibt eine Übersicht über die in einem AL-Programmiersystem realisierten Komponenten [6.10] [6.18]. Im Unterschied zur ursprünglichen Version von AL [6.5] sind die Komponenten Compiler (AL-Standardmodus), Testsystem (ALAID) und interaktive Komponente (POINTY) in einem System zusammengefaßt und durch Modusauswahl ansprechbar. Außerdem ist die Trajektorienberechnung in die Ausführungszeit verlagert, um auf variable Umweltbedingungen besser reagieren zu können.

6.3. Programmiersysteme von Industrierobotersteuerungen

6.3.1. Grundfunktionen eines Programmiersystems

In diesem Kapitel werden die Funktionen des Programmiersystems und das Zusammenwirken mit der Steuerung behandelt. Das Steuerungssystem realisiert Bewegungsabläufe gemäß einem vorgegebenen Programm und sorgt für das Zusammenspiel zwischen den Teilsystemen. Dabei sind die Operationen der Bewegungs-, Aktions- und Programmablaufsteuerung und ggf. der Verarbeitung von Sensorinformationen zu realisieren. Zur Programmierung stehen für diese Operationen relativ abgeschlossene funktionelle Module zur Verfügung. Sie gehören zur Grundausstattung des IR-Systems und werden in der Regel vom Hersteller der Steuerung mitgeliefert. Die Programmierung der Grundfunktionen erfolgt in interner Form und ist auf einen konkreten Roboter bzw. eine Klasse von ähnlichen Typen zugeschnitten.

Das Programmiersystem muß es dem Anwender gestatten, alle Funktionen des Steuerungssystems aufzurufen und in einer geeigneten Form das Bewegungs- und Aktionsprogramm aufzustellen. Programmiersysteme können in Abhängigkeit von dem Leistungsvermögen der Steuerung und dem Ausstattungsgrad mit Sensoren und Prozeßkoppel-

einrichtungen sehr unterschiedlich ausgebaut sein. Es lassen sich im wesentlichen folgende Funktionen unterscheiden:

- Auslösen von Bewegungen;
- Auslösen von Aktionen;
- Programmtest und -korrektur;
- Programmverwaltung (Erstellen, Editieren, Löschen, Archivieren);
- Beschreibung der Verarbeitung von Daten über die Umwelt;
- Beschreibung der Kommunikation mit anderen Steuerungssystemen;
- Kommunikation mit dem Bediener (Menütechnik, Bedienerführung).

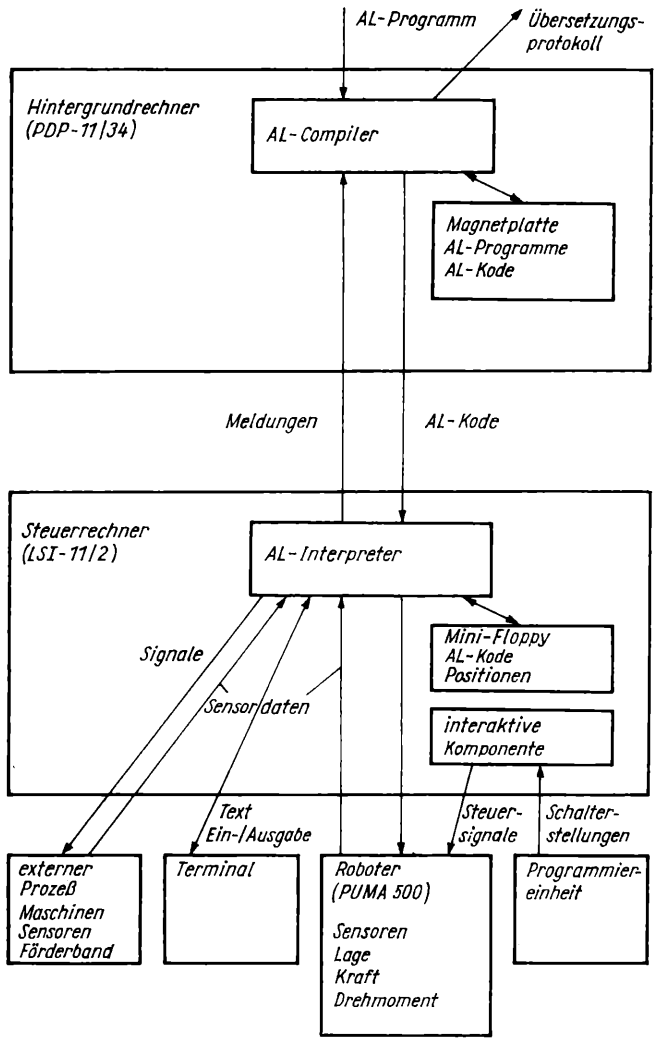


Bild 6.9
Übersicht über das Programmiersystem AL (nach [6.10])

6.3.2. Softwareentwicklung für Programmiersysteme

Bei den bisher eingesetzten Programmierverfahren für Industrieroboter werden die Erfahrungen der NC-Programmierung von Werkzeugmaschinen und der Anwendung von Teleoperatoren ausgenutzt. Unter Programmierung wird dabei nicht unbedingt die Auf-

stellung eines lesbaren (textuellen) Programms verstanden, sondern vorrangig die Festlegung eines Arbeitsablaufs in einem Speichermedium.

Seit der Entwicklung von Programmiersprachen für Industrieroboter und ihrer Implementierung auf Rechnersystemen zu Beginn der 70er Jahre ist die Notwendigkeit der Bereitstellung von komplexen Programmiersystemen für flexible Automatisierungslösungen bekannt (WAVE [6.4], AL [6.5], MAL [6.6]). An ihrer Realisierung wird intensiv gearbeitet, und zur Steuerung und Programmierung werden zunehmend leistungsfähige Softwaresysteme angeboten (z. B. VAL [6.7]).

Für die Realisierung von komplexen Softwaresystemen besteht aus Zeit- und Kostengründen die Forderung nach effektiven Programmiermethoden. Einige Grundprinzipien sind dabei

- Anwendung der strukturierten Programmierung mit den wesentlichen Kennzeichen der Modularität und der Portabilität der Programme,
- klare Definition von Modulschnittstellen (Bereitstellung von flexibel einsetzbaren Lösungsvarianten mit gleichen Schnittstellen),
- Generierbarkeit für angepaßte Lösungen (Modulauswahl),
- Integrationsfähigkeit in übergeordnete Systeme,
- Erweiterbarkeit des Funktionsumfangs,
- Verwendung höherer Programmiersprachen (z. B. PASCAL).

Bei Beachtung solcher grundlegenden Prinzipien ist die einmal für eine bestimmte Funktion erstellte Software in verschiedenen Systemen und auch bei Weiterentwicklung und Änderung der rechentechnischen Basis verfügbar, und unnötige Doppelarbeit wird vermieden.

Die Nachteile der Anwendung höherer Programmiersprachen (größerer Speicherbedarf und etwas längere Programmlaufzeiten) werden langfristig durch ständig sinkende Hardwarekosten für Speicherbausteine und Mikrorechnersysteme und durch die Wiederverwendbarkeit von Programmodulen in verschiedenen Systemen ausgeglichen. In den folgenden Abschnitten werden die Grundgedanken der Modularität und hierarchischen Gestaltung ausführlicher behandelt.

6.3.3. Modularer Aufbau von Steuerungssystemen

Das IR-Steuerungssystem (oder einfach Steuerung) realisiert das Zusammenspiel zwischen allen Teilsystemen des IR. Der Nutzer des Systems sieht die Steuerung lediglich als ein Hilfsmittel an, das mit Hilfe des Programmiersystems erstellte Programm abarbeiten zu lassen. Die Steuerung definiert damit eine untere Schnittstelle für die Kommunikation mit dem IR. Die interne Bearbeitung erfolgt als ein vom Programm gesteuerter Aufruf von einzelnen Teilfunktionen (Operationen).

Die im Abschnitt 6.3.2. genannten Grundforderungen der Modularität und Portabilität führen dazu, daß alle Teilfunktionen als relativ abgeschlossene Programmkomponenten (Module) mit einem genau definierten und im Vergleich zur Steuerungsaufgabe eingeschränkten Funktionsumfang realisiert werden. Diese Modularisierung bietet den Vorteil, daß überschaubare Programmteile entstehen, die bei Bedarf durch andere (neue Versionen oder vom Robotertyp abhängige Varianten) ausgetauscht werden können. Die Modularisierung zwingt auch zur Definition von Modulschnittstellen. Sind z. B. die Komponenten Achsregelung, Effektorsteuerung und Sensorsystem mit genormten Schnittstellen ausgerüstet, so lassen sich IR-Steuerungen generieren, bei denen Komponenten

unterschiedlicher Produzenten zusammengeschaltet werden können und die eigentliche Steuerungssoftware multivalent genutzt werden kann.

Da die Anforderungen an das Steuerprogrammsystem sehr stark vom Einsatz des Roboters abhängig sind, kommt (besonders bei prozeßflexiblen Anwendungsfällen) der Bereitstellung eines modular aufgebauten, auf ein bestimmtes Einsatzgebiet und einen konkreten Roboter abstimmbaren Steuerungssystems (Generierbarkeit) eine wachsende Bedeutung zu.

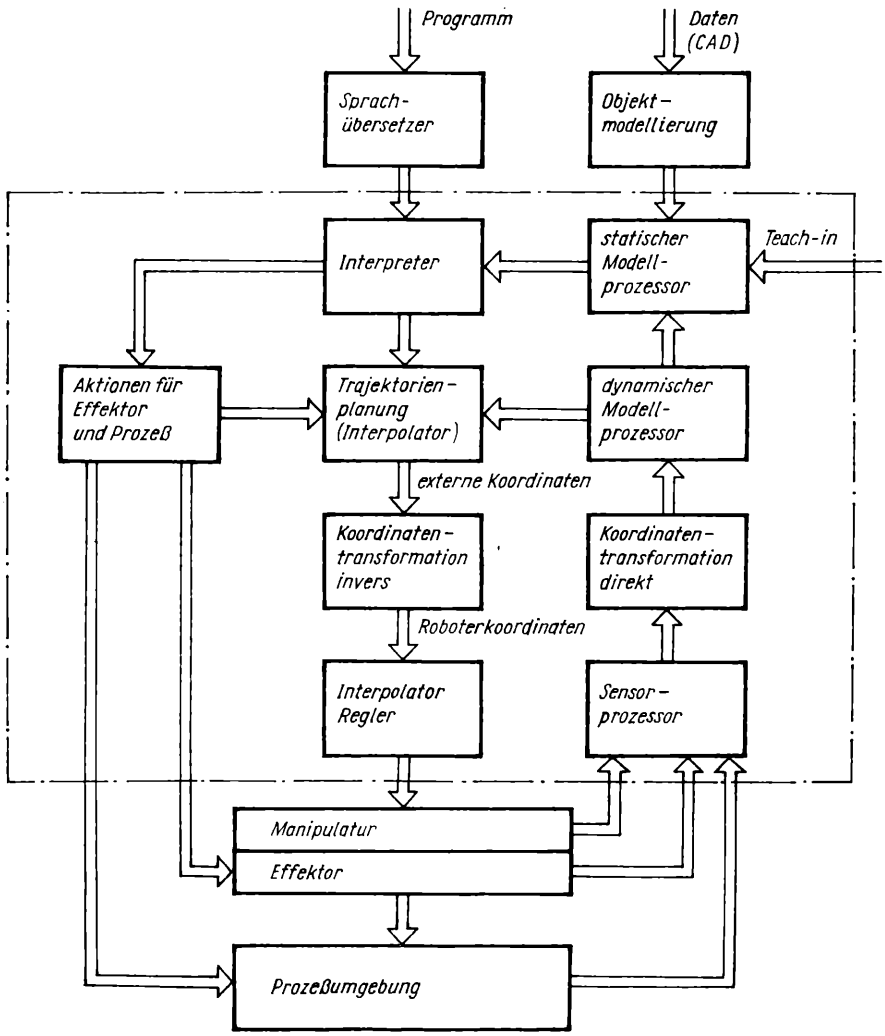


Bild 6.10. Grundstruktur eines modularen Steuerungs- und Programmiersystems

Die funktionelle Grundstruktur einer modularen IR-Steuerung, deren Programmierung im externen Koordinatensystem erfolgen kann, ist im Bild 6.10 dargestellt. Der Umfang der in den einzelnen Komponenten realisierten Algorithmen bestimmt die Leistungsfähigkeit der Steuerung. So kann z. B. die Koordinatentransformation für drei Freiheitsgrade (Position mit x, y, z) oder auch für sechs Freiheitsgrade (Position und Orientierung) realisiert sein. Die Berücksichtigung der Orientierung gestattet bei der zweiten Variante eine wesentliche einfachere Programmierung.

6.3.4. Hierarchische Gestaltung von Programmiersystemen

Die Integration von Industrierobotern in rechnergesteuerte Fertigungssysteme und der rechnergestützte Entwurf von Erzeugnissen (CAD *c*omputer-*a*ided *d*esign) machen den Einsatz hierarchisch aufgebauter Programmiersysteme mit zunehmender Komplexität möglich und notwendig. Die für den Industrieroboter erforderlichen Komponenten in einem solchen komplexen Programmier- und Steuerungssystem für zukünftige Fertigungsabschnitte sind im Bild 6.11 dargestellt.

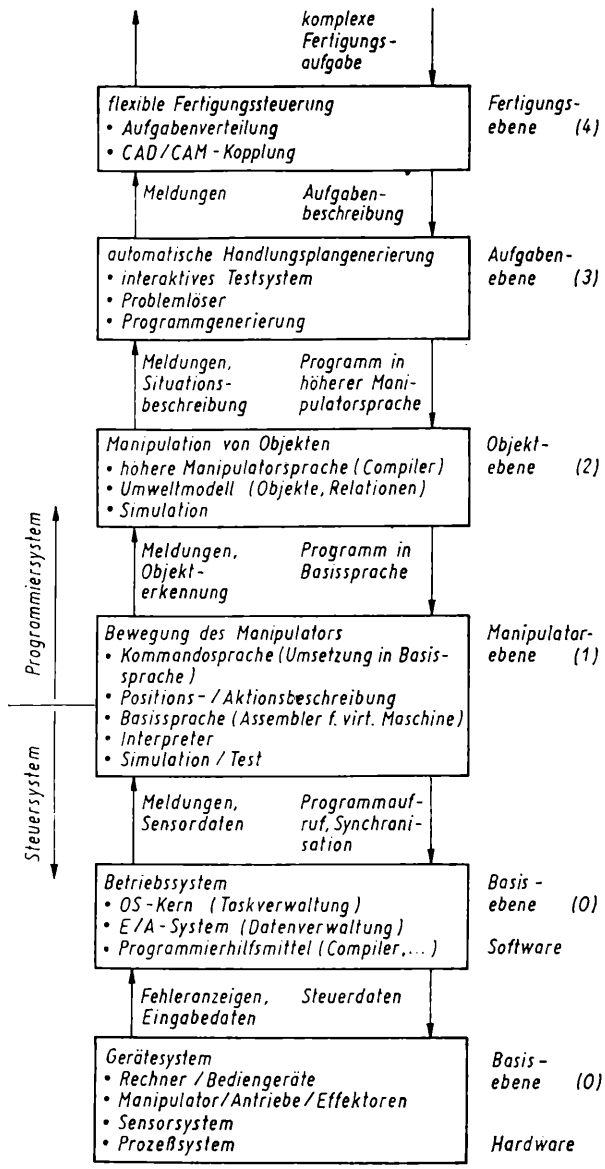


Bild 6.11 Hierarchieebenen der Industrieroboterprogrammierung in Fertigungssystemen

In der *Ebene der Fertigungssteuerung* ist der IR lediglich Teil einer komplexen Automatisierungsanlage und wird im Fertigungsprogramm als eine programmierbare Komponente mit bestimmten funktionellen Möglichkeiten angesehen. Die Aufgabenverteilung

realisiert ein Maschinenbelegungsprogramm, das die momentane Verfügbarkeit sowie bestimmte Optimierungskriterien des Fertigungsablaufs berücksichtigt. Die benötigten Objektdaten werden direkt aus einer CAD-Datei gewonnen.

Die *Aufgabenebene* erzeugt aus einer problemorientierten Aufgabenformulierung ein IR-Programm auf dem Objektniveau. Die Aufgabe des Programmiersystems besteht auf diesem Niveau darin, durch Nutzung von Wissen über den technologischen Prozeß eine geeignete Ablauffolge in Form einer Objekt- und Handlungsbeschreibung in der höheren Manipulatorsprache zu erzeugen. Für einfache Aufgabenstellungen ist zur automatischen Handlungsplangenerierung ein System von vorgefertigten Makrobausteinen ausreichend, die für konkrete Fälle zu einem Programm zusammengefügt werden. Bei komplexeren Aufgabenstellungen werden Verfahren der künstlichen Intelligenz (z. B. Problemlöser) eingesetzt, die aus einer sehr kurzen Aufgabenformulierung ein komplettes Programm aufstellen können.

Die *Objektebene* realisiert die Umsetzung des IR-Programms in Form einer Objekt- und Aktionsbeschreibung in ein Programm zur Beschreibung von IR-Funktionen. Dabei sind die konkreten Umweltrestriktionen anhand eines Umweltmodells zu berücksichtigen. Durch ein ausgebautes Simulationssystem lassen sich auch die geometrischen Abmessungen des Manipulators und dessen dynamisches Verhalten einbeziehen. Das entstehende Programm ist damit vom konkreten Robotertyp abhängig.

Auf der *Manipulatorebene* wird für jede IR-Funktion ein Aufruf von Komponenten des Steuerungssystems realisiert. Diese Ebene stellt als virtuelle Maschine die Verbindung des Programmiersystems mit dem Steuerungssystem dar. Bezüglich des Programmiersystems lassen sich alle Funktionen des Steuerungssystems, des Betriebssystems und der Hardware in einer Basisebene zusammenfassen, die sich für den Anwender als nicht frei programmierbar darstellt.

Werden Industrieroboter im technologischen Fertigungsprozeß eingesetzt, erfüllen sie eine spezielle Teilaufgabe in einem komplexen Zusammenhang. In jeder der Hierarchieebenen findet eine Aufgabendekomposition in eine überschaubare Anzahl von Operationen statt. Diese hierarchische Aufgabendekomposition wird in ein hierarchisches Programmiersystem übertragen, um den Zugriff auf die unterschiedlichen Ebenen zu gewährleisten. Da die Komplexität der Aufgaben mit steigendem Abstraktionsgrad zunimmt, ist zur Realisierung des Programmiersystems ein Rechnersystem erforderlich [6.2]. Die Programmierung des IR erfolgt dabei z. B. auf einem Kleinrechner (Programmentwicklung) und die Programmabarbeitung (Steuerung) auf einem Mikrorechnersystem.

7. Sensoren

Die für einen Handlungszyklus des Industrieroboters erforderlichen Bewegungsdaten können auf drei unterschiedliche Arten gewonnen werden:

- durch direktes oder indirektes Teach-in, wobei der Roboter selbst oder ein geeignetes Hilfsgerät zur Digitalisierung von Raumpunkten benutzt wird;
- aus einem einmal erstellten Umweltmodell durch Off-line-Programmierung, wobei die Bewegungsdaten anhand eines Datenspeichers gewonnen werden;
- durch ein Grobmodell der Umwelt, das durch äußere Sensoren vervollständigt und aktualisiert wird.

Die ersten beiden Methoden setzen voraus, daß das der Datengewinnung zugrunde gelegte Umweltmodell statisch ist oder sich nur in determinierter Weise ändert.

Hieraus resultieren erhebliche Genauigkeitsforderungen an den technologischen Ablauf, die im wesentlichen in der Herstellung exakt reproduzierbarer Zustände durch periphere Einrichtungen (Paletten, Fördergeräte, Spanntische) bestehen. Eine Veränderung der Arbeitsaufgabe ist dabei in der Regel mit aufwendigen mechanischen Umrüstungen, Justierung und Erstellung neuer Bewegungsprogramme verbunden.

Mit steigenden Forderungen an die Genauigkeit wächst der Aufwand für Roboter, periphere Einrichtungen und Programmierung erheblich.

Daneben gibt es zahlreiche technologische Prozesse, bei denen Werkstücktoleranzen, indeterminierte Positionen und unbekannte Werkstücksequenzen auftreten, die den Einsatz von Industrierobotern mit festprogrammierten Bewegungsdaten ausschließen.

Durch die Entwicklung und Anwendung adaptiver Robotersysteme können die Forderungen an die Genauigkeit und die Arbeitsvorbereitung herabgesetzt und effektive Einsatzmöglichkeiten im Bereich kleinerer Losgrößen geschaffen werden.

Adaption bedeutet in diesem Zusammenhang, daß anhand eines groben Umweltmodells Abläufe und Bewegungsdaten programmiert und aktuell auftretende Zustandsänderungen zur Korrektur des programmierten Ablaufs herangezogen werden. Die Änderung der Umweltparameter wird dabei durch Sensoren erfaßt.

Diese Sensoren werden im Gegensatz zu den inneren Meßsystemen des IR, die den Zustand des kinematischen Systems erfassen, als äußere Sensoren bezeichnet.

Äußere Sensoren sind dabei nicht nur zur Erfassung von Prozeßparametern in der Arbeitsphase, sondern auch zur Vereinfachung des Teach-in (Kollisionsverhütung, Genaupositionieren) und zur Gewinnung konstanter Umweltparameter (Modellbildung) geeignet. Äußere Sensoren können aber auch bei relativ einfachen Steuerungsaufgaben entscheidend zur Verbesserung der Einsatzparameter beitragen oder einige Einsatzgebiete überhaupt erst ermöglichen.

In diesem Abschnitt sollen anhand ausgewählter Beispiele die Wirkprinzipien äußerer Sensorsysteme, ihre potentiellen Einsatzmöglichkeiten und die Entwicklungstendenzen dargestellt werden.

7.1. Äußere Sensoren und Steuerungsfunktionen

Sensoren dienen der Erfassung relevanter physikalischer Größen im Arbeitsbereich des Industrieroboters, um ihn gemäß der gestellten Aufgabe zu einer sinnvollen Reaktion auf Veränderungen im Arbeitsraum zu befähigen. Eine Übersicht zu den erforderlichen Meßaufgaben bei unterschiedlichen Aktionen des Roboters gibt Tafel 7.1.

Tafel 7.1. Meßaufgaben von Sensorsystemen bei einigen IR-Operationen

Operation des IR	**Meßaufgabe**
Punkt-zu-Punkt-Bewegung	Erfassung von Hindernissen (Kollisionsschutz) Messung der Zielkoordinaten zum Genaupositionieren
Greifen definierter Objekte mit Positionsabweichung	Anwesenheit des Objektes Zentrieren des Greifers in der Greifebene
Greifen zerbrechlicher Objekte	Messung der Greifkräfte
Bewegung auf technologisch bedingten Bahnen	Messung toleranzbedingter Abweichungen von Referenzbahnen absolute Bahnvermessung
Kraftschlüssige technologische Operationen (Fügen, Schleifen, Polieren)	Messung der ausgeübten Kräfte oder Reaktionskräfte
Greifen undefinierter Objekte	Erkennung und Klassifizierung Lagebestimmung
Bewegung in undefiniertem oder zeitlich variablem Arbeitsraum	Erkennung und Klassifizierung von Objekten und Relationen zwischen Objekten

Dabei ist zu unterscheiden zwischen Aufgaben, die auch ohne Sensoren prinzipiell lösbar sind, wo durch ihren Einsatz aber Zeiteinsparung bei der Programmierung oder eine Herabsetzung der Forderungen an die Genauigkeit der Peripherie möglich ist, und solchen Aufgaben, die ohne Sensoren nicht durchführbar sind. Zu der zweiten Klasse von Aufgaben rechnen insbesondere Fügevorgänge, für die die Wiederholgenauigkeit der Industrieroboter beim Positionieren nicht ausreichend ist oder Toleranzen beim Fügen zu berücksichtigen sind. Beim Lichtbogenschweißen werden in der Regel nur geringe Forderungen an die Genauigkeit der Werkstücke gestellt; die Notwendigkeit des Einsatzes von Sensoren resultiert in erster Linie aus der Führung des Schweißprozesses selbst und dem damit verbundenen thermischen Versatz.

Typisch für Industrierobotersensoren ist, daß aufgrund der Bewegungsmöglichkeiten und des Vorhandenseins eines inneren Meßsystems die benötigten Informationen auf sehr unterschiedliche Weise gewonnen werden können. Positionen von Werkstücken im Arbeitsraum können beispielsweise durch taktilen Kontakt des Effektors mit dem Werkstück und Auswertung der Lagemeßwerte des kinematischen Systems, durch Messung relevanter Distanzen zwischen Effektor und Werkstück oder durch Auswertung des mittels eines Erkennungssystems gewonnenen Abbilds des Arbeitsraums gewonnen werden.

Hinzu kommt, daß die Umwelt des Industrieroboters als Teil eines technologischen Gesamtprozesses in weiten Grenzen gestaltbar ist. Markierungen an Werkstücken, die ihre Funktion nicht beeinträchtigen, können beispielsweise sehr effektiv zur Unterscheidung verschiedener Werkstücke mit sehr einfachen Sensoren dienen.

Äußere Sensoren müssen die benötigte Information in einer von der Steuerung verarbeitbaren Form liefern. Die möglichen funktionellen Schnittstellen zur Steuerung sind im Bild 7.1 dargestellt. Der direkte Zugriff auf die Lageregler gestattet die kürzesten Reaktionszeiten und ist somit besonders zur Kollisionsverhütung und zum Überlastschutz geeignet. Liegen die Meßwerte in den Koordinaten der einzelnen Achsen vor, läßt sich deren Bewegungsverlauf durch Einspeisung in die vor der Gelenkinterpolation liegende Schnittstelle 2 beeinflussen. Korrekturen des Bahnverlaufs und der Geschwindigkeit müssen vor der inversen Koordinatentransformation (Schnittstelle 3) vorgenommen werden. Die Schnittstelle 4 gestattet die Auswahl unterschiedlicher Bahnend- und Zwischenpunkte und Bahnsegmente. Durch die Schnittstelle 5 werden unterschiedliche Bewegungsprogramme aus dem Arbeitsspeicher ausgewählt.

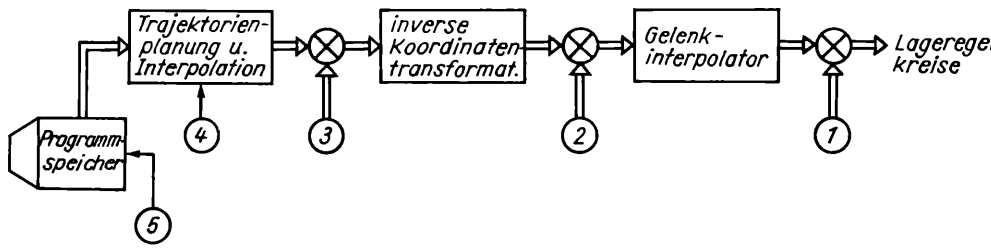


Bild 7.1. Funktionelle Schnittstellen zur adaptiven Steuerung von Industrierobotern

Die Sensorinformationen lassen sich grob unterteilen in

- Bestimmung der Anwesenheit von Objekten oder Objektteilen in einem gegebenen Bereich,
- Bestimmung räumlicher Relationen zwischen Effektor und Werkstück oder zwischen Objekten bzw. Objektteilen,
- Messung von Wechselwirkungskräften zwischen Industrieroboter und Umwelt,
- Identifikation von Werkstücken oder Teilen von Werkstücken.

Hinzu kommen weitere Meßaufgaben zur Funktionssicherung, Diagnose, Bestimmung von Qualitätsparametern usw., die nicht roboterspezifisch sind und die bei allen Automatisierungsaufgaben in ähnlicher Weise zu lösen sind.

Für jede der Meßaufgaben sind mehrere Meßprinzipien und unterschiedliche Primärwandler verwendbar. Ihre Eignung hängt dabei wesentlich von den technologischen Randbedingungen und den konkreten Störeinflüssen ab.

Nach der Art der Informationsgewinnung werden äußere Sensoren für Industrieroboter in

- taktile Sensoren,
- Näherungssensoren und
- Erkennungssysteme

eingeteilt.

Bei den Sensoren unterscheidet man außerdem solche, die eine skalare Größe, eine vektorielle Größe oder skalare bzw. vektorielle Größen in ein bis drei Dimensionen aufnehmen. Je komplexer die aufgenommene Information ist, um so aufwendiger wird einerseits die zur Ableitung der Steuerungsgröße benötigte Informationsgewinnung, um so flexibler lassen sich aber andererseits die Sensoren für unterschiedliche Aufgaben einsetzen.

Eine besondere Bedeutung kommt dabei den visuellen Erkennungssystemen zu, da mit ihnen ganze Umweltausschnitte mit hoher Auflösung erfaßt werden können.

Ähnliche Forderungen werden zukünftig auch taktile Matrixsensoren zu erfüllen haben, die aus dem formschlüssigen Kontakt Werkstücke identifizieren und deren Lage im Greifersystem ermitteln müssen. Sie werden insbesondere dann eingesetzt, wenn die Bedingungen für den Einsatz visueller Erkennungssysteme nicht gegeben sind, z. B. da Werkstückteile durch den Greifer verdeckt sind oder keine ausreichende Beleuchtung möglich ist.

Um die Informationen aus dem unmittelbaren Arbeitsbereich des Industrieroboters zu erfassen, ist eine Anbringung in unmittelbarer Nähe des Effektors erforderlich. Dies ist auch eine wichtige Voraussetzung für ihren flexiblen Einsatz [7.1]. Um eine störsichere Übertragung in die Steuerung zu gewährleisten, ist dabei eine Vorverarbeitung und Verdichtung der von den Primärwandlern gelieferten Signale vorzunehmen.

7.2. Taktile Sensoren

Mit Hilfe berührender Sensoren werden zwei Arten von Informationen gewonnen, die sich in ihrer Qualität unterscheiden. Erstens werden durch Herstellung eines Kontakts zwischen einem Meßfühler und einem zu vermessenden Objekt Informationen über die Anwesenheit und die Form des Objekts gewonnen; die Wechselwirkung zwischen Objekt und Meßfühler wird dabei so gering wie möglich gehalten. Zweitens werden taktile Sensoren zur Messung der Kräfte und Momente eingesetzt, die bei der Wechselwirkung des Roboters mit dem zu hantierenden oder zu bearbeitenden Objekt auftreten. Da die direkte Messung von Kräften nicht möglich ist, erfolgt in beiden Fällen die Messung von Längenänderungen eines deformierbaren Körpers.

7.2.1. Taktile Meßfühler

Das Prinzip taktiler Meßfühler ist im Bild 7.2 dargestellt. Ein elastisch aufgehängter Tastfinger wird bei Kontakt mit einem Objekt ausgelenkt. Die Meßwandler können entweder als Kontaktschalter oder als Distanzwandler ausgeführt sein; man unterscheidet danach schaltende und messende Tastsensoren.

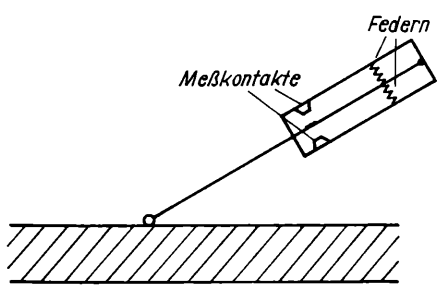


Bild 7.2
Taktiler Meßfühler

Schaltende Tastsensoren mit einem Schaltbereich von einigen zehnteln Millimetern werden in Verbindung mit dem inneren Meßsystem des IR zur groben Vermessung von Objektflächen, Konturverfolgung und Abstandsführung (z. B. beim Schweißen) eingesetzt.

Durch Kombination eines hochauflösenden inneren Meßsystems mit einem messenden

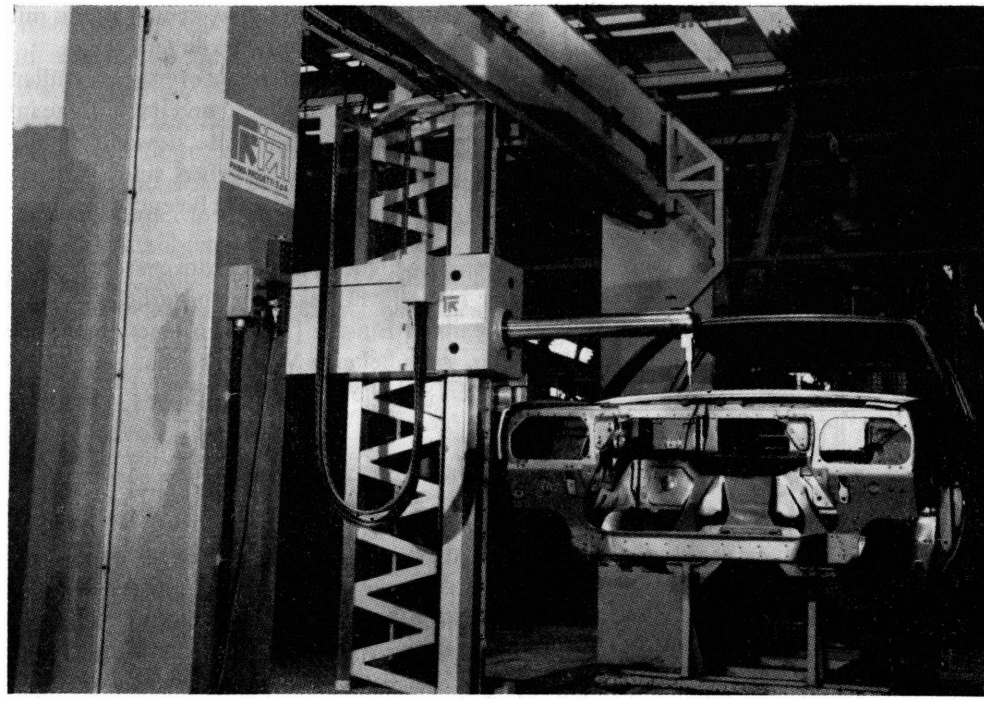


Bild 7.3. Meßroboter zur Bestimmung von Formabweichungen bei Autokarrosserien (Werkfoto Prima Progetti Spa, Turin)

Tastfühler entstehen Meßroboter oder Mehrkoordinatenmeßsysteme (Bild 7.3). Dabei werden durch definierte Bewegungen des kinematischen Systems Kontaktpunkte mit dem Objekt ermittelt, deren Positionen mit Referenzdaten verglichen werden. Die eingesetzten Tastsensoren messen dabei in bis zu drei Koordinatenachsen. Als Wegaufnehmer finden induktive Meßwandler Verwendung.

7.2.2. Kraft-Moment-Sensoren

Die Erfassung äußerer Kräfte und Drehmomente ist zur definierten Steuerung der Reaktionskräfte bei Montageaufgaben, Bearbeitungsverfahren wie Gußputzen, Entgraten, Polieren sowie zur Überlasterkennung und Schadensverhütung bei Kollisionen notwendig. Bewegungsbahnen und -geschwindigkeiten sind dabei durch die gemessenen Kräfte und Momente so zu adaptieren, daß vorgegebene Reaktionskräfte eingehalten werden. In den genannten Einsatzgebieten treten gegenüber dem beim Programmieren vorliegenden Umweltmodell durch Werkstücktoleranzen, Aufspannabweichungen, Werkzeugverschleiß und technologisch bedingte stochastische Konturverläufe (z. B. Angußreste, Grat) Abweichungen auf, die in der Bearbeitungsphase zu berücksichtigen sind. Im einzelnen werden durch die Nutzung von Kraft-Moment-Messungen zur Steuerung des Industrieroboters folgende Einsatzgebiete erschlossen:

Fügen und Montieren

Die zu fügenden Teile sind so zu positionieren, daß keine Verklemmung auftritt; die Fügekraft ist zu regeln; toleranzbedingte Überlastung ist zu vermeiden. (Auf die für spezielle Anwendungsfälle entwickelten, passiven und selbstjustierenden Fügemechanismen

wird im folgenden nicht eingegangen; hier sei auf ausführliche Beschreibungen in [7.3] [7.4] verwiesen.)

Schleifen und Entgraten

Die Arbeitsgeschwindigkeit des Industrieroboters ist so zu regeln, daß er im Bereich seiner eigenen Nennbelastung bzw. der des Werkzeugs arbeitet. Toleranzbedingte Bahnabweichungen und Werkzeugverschleiß sind auszugleichen.

Beschicken

Regelung von Greifkräften bei zerbrechlichen Gegenständen, Erkennen von Überlastungen bei Kollision oder Klemmen von Werkstücken.

Prinzipiell sind drei Varianten der Kraft- und Momentmessung möglich:

- Ermittlung über die Antriebsrückwirkung,
- Erfassung mit Sensoren in einem Bearbeitungstisch,
- Erfassung mit Sensoren am Handgelenk in der Nähe des Werkzeugs oder Greifers.

Die erste Möglichkeit hat sich als zu ungenau erwiesen, da in die Transformationsbeziehungen zur Kraftgrößenermittlung am Kraftangriffspunkt das gesamte Robotermodell einschließlich der Antriebscharakteristik und nichterfaßbarer Reibungen einfließen.

Eine Ausnahme bildet die Bearbeitung von Werkstücken durch Entgraten, wenn das Werkstück vom Roboter an der Schleifmaschine geführt wird. Die Messung der Leistungsaufnahme der Schleifeinrichtung kann hier grob zur Steuerung der Verfahrgeschwindigkeit genutzt werden.

Die Messung der Kräfte und Momente im Bearbeitungstisch führt in der Regel zu speziellen Peripherielösungen und erfordert hohen Umrüstaufwand bei veränderten Aufgaben.

International durchgesetzt haben sich aus diesen Gründen Sensoren, die in den Roboter integriert werden können und damit die flexibelsten Einsatzmöglichkeiten gestatten.

Die adaptive Bewegungssteuerung eines Roboters mit fünf und mehr Freiheitsgraden erfordert die Messung aller sechs mechanischen Schnittgrößen im Koordinatensystem des Effektors.

Mechanische Schnittgrößen (Kräfte, Momente) werden durch Einstellung eines Gleichgewichtszustands mit einer der Schnittgröße entgegengerichteten Kraft- oder Momentengröße ermittelt. Diese Gegenschnittgröße wird in der Regel durch Deformation eines elastischen Verformungskörpers erzeugt. Die Größe der Verformung dient bei entsprechender Eichung als Maß für die Schnittgröße. Man unterscheidet elektrisch aktive und elektrisch inaktive Verformkörper. Bei elektrisch inaktiven Verformkörpern wird die Deformation durch Dehnungsmessung oder Distanzmessung ermittelt. Beide Prinzipien sind im Bild 7.4 am Beispiel der Greifkraftmessung dargestellt.

Dabei wird der lineare Zusammenhang zwischen Kraft und Deformation genutzt (Hookesches Gesetz).

Beide Meßmethoden werden auch bei der Konstruktion von 6-Komponenten-Sensoren angewendet.

Da es schwierig und mechanisch zu aufwendig ist, für jede Kraft- oder Momentkomponente einen separaten Verformkörper zu installieren und ihre Angriffspunkte so zu gestalten, daß die Schnittkräfte in einem gemeinsamen Koordinatensystem gemessen werden, sind Verformkörper so konstruiert, daß nicht die Kraft- und Momentkomponenten direkt, sondern Linearkombinationen der einzelnen Beiträge bestimmt werden.

Ein Beispiel hierzu wird im Bild 7.5 gezeigt. Der Verformkörper enthält acht Biegestäbe (P_x, P_x^+, P_v, P_v^+, Q_x, Q_x^+, Q_y, Q_y^+), auf denen je zwei Dehnmeßstreifen angebracht sind. Das Dehnmeßstreifenpaar ist Bestandteil einer temperaturkompensierten Meßbrücke, die ein verformungsanaloges Ausgangssignal liefert.

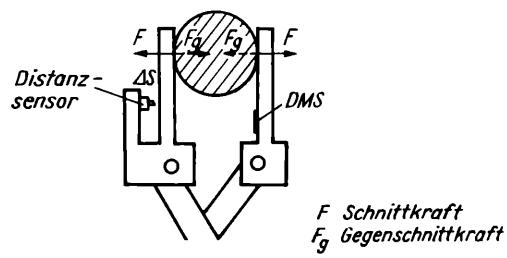


Bild 7.4. Prinzip der Deformationsmessung mit Distanzwandler und Dehnmeßstreifen

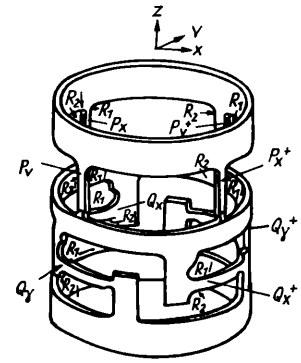


Bild 7.5. Verformkörper mit acht Dehnmeßstreifenpaaren zur Kraft- und Momentenmessung

Für den Zusammenhang zwischen den Ausgangsspannungen der Aufnehmer und den Kraft-Moment-Komponenten, bezogen auf ein festes Koordinatensystem im Verformkörper, gilt

$$U_i = c_{i1}f_x + c_{i2}f_y + c_{i3}f_z + c_{i4}m_x + c_{i5}m_y + c_{i6}m_z$$

oder

$$\begin{bmatrix} U_1 \\ \vdots \\ U_8 \end{bmatrix} = \begin{bmatrix} c_{11} & \dots & c_{16} \\ \vdots & & \vdots \\ c_{81} & & c_{86} \end{bmatrix} \begin{bmatrix} f_x \\ f_y \\ f_z \\ m_x \\ m_y \\ m_z \end{bmatrix} = \boldsymbol{CF}. \tag{7.1}$$

$\boldsymbol{C}$ ist die sog. Koppelmatrix; ihre Elemente werden in der Regel durch definierte Kräfte und Momente in den Koordinatenachsen ermittelt. Zur Berechnung der Kräfte ist entsprechend

$$\boldsymbol{F} = \boldsymbol{C}^{-1}U \tag{7.2}$$

die Inverse der Koppelmatrix zu bilden. Da die Inverse nur für quadratische Matrizen gebildet werden kann, gilt hier

$$\boldsymbol{F} = (\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C})^{-1}\,\boldsymbol{C}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{C}^{\mathrm{I}}\boldsymbol{U} \tag{7.3}$$

mit C^{I} als Pseudoinverse.

Die Invertierbarkeit der Matrizen C bzw. $C^{\mathrm{T}}C$ ist nur gesichert, wenn die Anordnung der Meßwandler und der meßtechnisch aktiven Verformkörperteile so gewählt ist, daß die Ausgangssignale linear unabhängig sind.

Wird die Anordnung und Gestaltung der meßtechnisch aktiven Zonen und der Meßwandler so gewählt, daß eine direkte Zuordnung zwischen einer Kraftgrößenkomponente und einem einzigen elektrischen Ausgangssignal erzielt wird, ist C eine Diagonalmatrix.

Bei der einfachen und preiswerten Gestaltung von mehrkomponentigen Sensoren werden zwei Wege beschritten. Der erste besteht darin, den Verformkörper so einfach wie möglich zu gestalten und die zahlreichen auftretenden Verkopplungen durch erhöhten Rechenaufwand bei der Dekomposition der Komponenten zu berücksichtigen. Ein Beispiel wird im Bild 7.6 gezeigt. An den drei meßtechnisch aktiven Stegen sind je zwei Dehnmeßstreifenpaare angeordnet; jedes Paar mißt eine Längsdehnung oder Scherkraft.

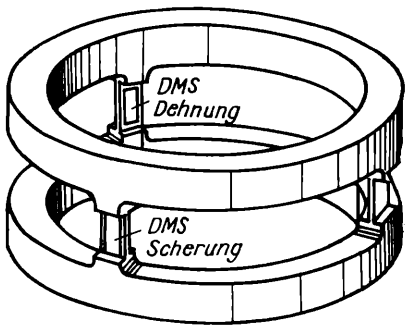


Bild 7.6
Verformkörper mit drei Stegen (nach [7.9])

Nachteilig wirkt sich dabei aus, daß für die Verformung Vorzugsrichtungen auftreten, in denen der Verformkörper bei gleicher Kraft unterschiedlich stark deformiert wird.

Der zweite Weg besteht in der Konstruktion von Verformkörpern, bei denen die Entkopplungsmatrix C möglichst viele Nullelemente besitzt. Dies kann durch zusätzliche Verformelemente geschehen; das Prinzip ist im Bild 7.7 dargestellt [7.5].

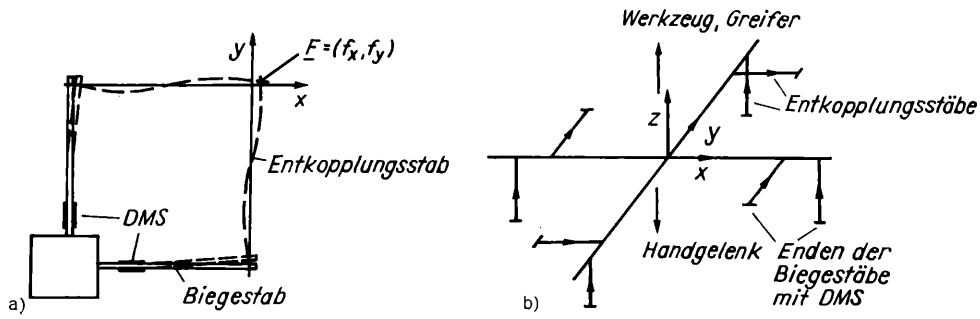


Bild 7.7. a) Prinzip der mechanischen Entkopplung von Verformelementen und b) Anordnung der Entkopplungsstäbe im Kraft-Momenten-Sensor

Unterschiedliche Vorspannungen der Verformteile und Eigengewicht von Sensor- und Effektorteilen führen zu zusätzlichen additiven Komponenten des Ausgangsspannungsvektors U, der bei der Auswertung gemäß

$$F = C^{\mathrm{I}}(U - B) \tag{7.4}$$

zu berücksichtigen ist.

Der Biasvektor B faßt alle additiven Übertragungsfehler zusammen und kann zur Unterdrückung unerwünschter konstanter Kräfte durch Eichung ermittelt werden. Ein Beispiel für die Ausführung von Kraft-Moment-Sensoren mit Abstandsmeßwandlern wird im Bild 7.8 gezeigt. Das Innenteil der Meßdose ist durch Federn so mit dem Außenring verbunden, daß es in allen Richtungen beweglich ist und möglichst richtungsunab-

hängige Gegenkräfte überträgt. Die Meßwandler 1 bis 3 sind in unterschiedlichem Winkel zum Boden des Außenrings angebracht, damit linear unabhängige Meßgrößen erfaßt werden.

Als Meßwandler kommen induktive, kapazitive oder optoelektronische Wandler (s. Abschn. 7.3.) in Frage, wobei eine Zwischenübertragung durch mechanische Stifte vorgenommen wird.

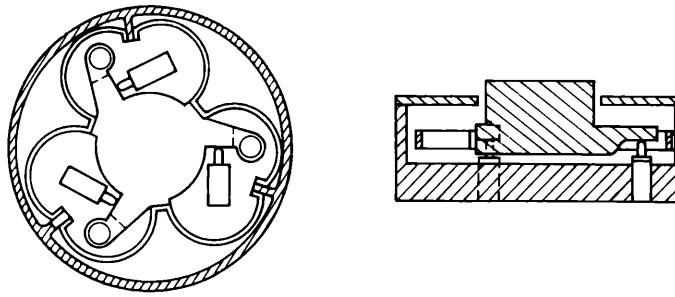


Bild 7.8
Kraft-Moment-Sensor mit sechs Distanzwandlern
(nach [7.8])

Wichtige Kenngrößen für Kraft-Moment-Meßeinrichtungen sind

- Verformung bei Nennkraft – sie sollte 0,1 bis 0,2 mm nicht überschreiten, da sie die Positioniergenauigkeit der vor der Dose liegenden Greifer oder Werkzeuge beeinflußt;
- Überlastsicherheit – Kräfte bis zum 10fachen der Nennlast dürfen den Verformkörper nicht zerstören; dies kann durch mechanische Anschläge erreicht werden, damit sind die Meßwandler geschützt;
- geringe Hysterese des Verformteils;
- geringe Eigenmasse.

Im Bild 7.9 ist die Struktur der Signalverarbeitung für einen Kraft-Moment-Sensor mit sechs Dehnmeßstreifenbrücken als Primärwandler dargestellt.

Die Brückenausgangssignale werden nach der Vorverstärkung tiefpaßgefiltert, um Einflüsse von Materialschwingungen im Verformkörper und im mechanischen Teil des Industrieroboters zu eliminieren. Für den erforderlichen Genauigkeitsbereich und die Auflösung der Primärwandler ist in der Regel eine A/D-Wandlung in 100 Amplitudenstufen ausreichend; das entspricht mit Vorzeichen 8 Bit. Nach jedem Einlesen eines Meßwerts wird zunächst ein Überlasttest durchgeführt. Wird ein vorgegebener Grenzwert überschritten, erfolgt sofort eine Interruptmeldung, die über einen speziellen Ausgang in den Steuerrechner gelangt und zum sofortigen Stop der Antriebe genutzt werden kann. In einem weiteren Digitalfilter, das programmtechnisch realisiert ist, wird die für spezifische Aufgaben benötigte Tiefpaßcharakteristik programmiert. Nach Subtraktion des Biasvektors erfolgt die Entkopplung. Anschließend wird mit einem Kalibrierprogramm der Kraft-Moment-Vektor in das Effektorkoordinatensystem umgerechnet.

Wegen des hohen Anteils der Multiplikationen (in der Entkopplungsmatrix bis zu 48) wird zur Echtzeitverarbeitung häufig ein spezieller Arithmetikprozessor eingesetzt. Damit sind Verarbeitungszeiten für einen vollständigen Kraft-Moment-Vektor von 10 ms erreichbar.

Kraft-Moment-Sensoren werden für unterschiedliche Nennbelastung industriell gefertigt, z.T. mit vollständiger Vorverarbeitung und Mikrorechner innerhalb der Meßdose [7.6]. Im Bild 7.10 ist der Einsatz eines Kraft-Moment-Sensors beim Schleifen von Angußresten schematisch dargestellt. Zur Erzielung einer maximalen Arbeitsgeschwindigkeit ist die Verfahrgeschwindigkeit des IR so zu regeln, daß die Schleifkraft $\boldsymbol{P}$, die sich aus drei Komponenten zusammensetzt – die zur Schleifscheibe tangentiale Komponente ist

nicht eingezeichnet –, einen möglichst konstanten Wert hat. Durch die Werkstücktoleranzen und den Schleifkörperabrieb ist außerdem eine Trajektorienanpassung in x-Richtung notwendig. Ist die Gratabmessung in der z-Richtung wesentlich kleiner als der Schleifscheibendurchmesser, so ist eine gleichzeitige Adaption der Trajektorie und der Verfahrgeschwindigkeit möglich, da die Kräfteänderungen durch Abrieb und Toleranz wesentlich langsamer verlaufen als die durch den Grat bewirkten Änderungen [7.7].

In der Montage werden Kraft-Moment-Sensoren eingesetzt, um das Verklemmen von

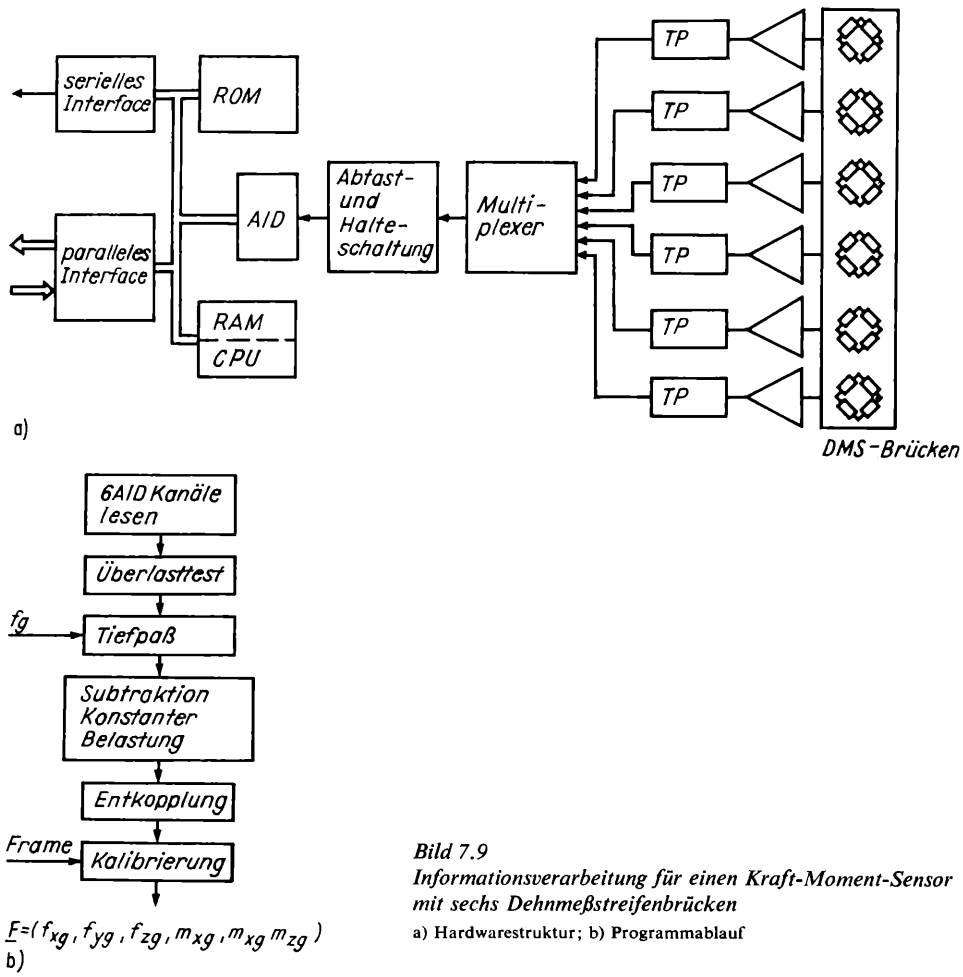


Bild 7.9
Informationsverarbeitung für einen Kraft-Moment-Sensor mit sechs Dehnmeßstreifenbrücken
a) Hardwarestruktur; b) Programmablauf

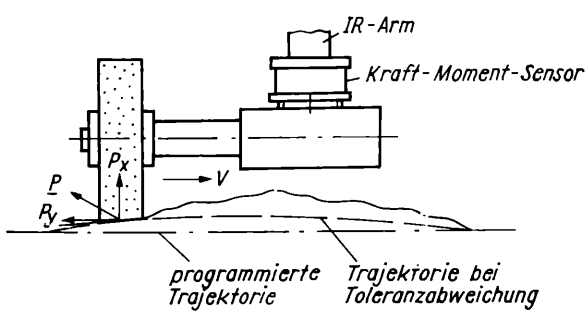


Bild 7.10
Vorschubregelung und Trajektorienanpassung beim Schleifen

Fügeteilen beim Justieren durch eine Gegenbewegung aufzuheben und die Fügekraft zu regeln.

Werden die hierzu erforderlichen Bewegungsgenauigkeiten vom IR nicht gewährleistet, kommen zur Justage spezielle Montagehilfseinrichtungen zur Anwendung, die sehr präzise Ausgleichsbewegungen durchführen [7.3].

7.2.3. Taktile Matrixsensoren

Zur groben Form- und Lageerkennung können matrixförmig angeordnete binäre oder analoge Berührungssensoren in den Greifflächen von Industrierobotern eingesetzt werden. Die einfachste Lösung besteht aus einem Rasterfeld von Druckschaltern.

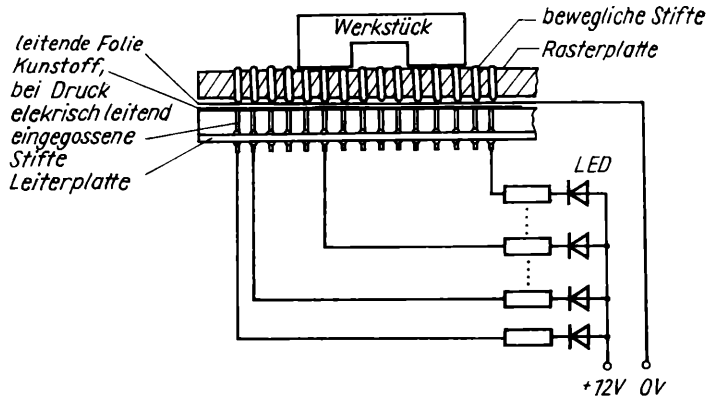
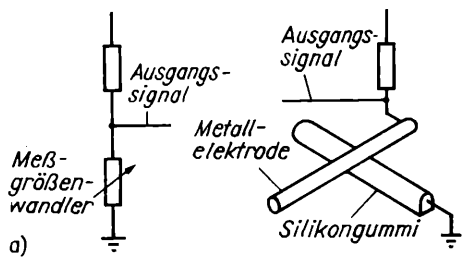
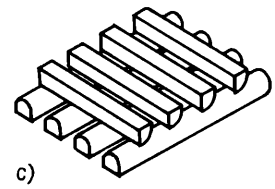
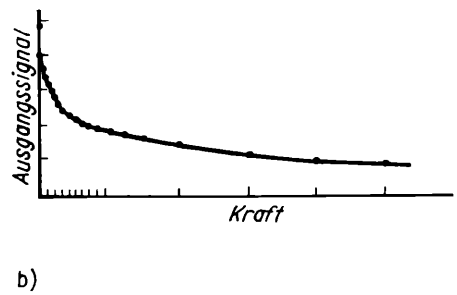


Bild 7.11
Taktile Matrix mit beweglichen Stiften







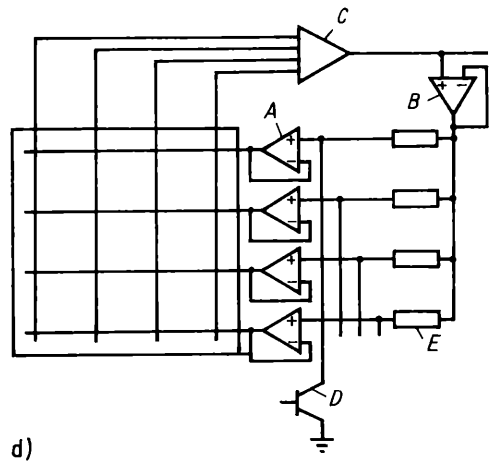


Bild 7.12. Taktile Matrix (nach [7.10])

a) Wandlungsprinzip; b) Übertragungskennlinie; c) geometrische Anordnung; d) Abtastung und Signalvorverarbeitung

Bild 7.11 zeigt ein Beispiel für eine analog arbeitende Druckmatrix. Bewegliche Stifte drücken bei Werkstückkontakt eine leitende Folie auf eine Kunststoffolie, wobei der Übergangswiderstand zwischen der leitenden Folie und den eingegossenen Stiften stark verringert wird. Die Auskopplung der einzelnen Signale kann über Optokoppler erfolgen. Die prinzipiellen Verfahren zur Auswertung solcher Matrixinformationen sind mit den Methoden der visuellen 2-D-Erkennung, die im Abschnitt 7.4. erläutert werden, identisch und sollen hier nicht behandelt werden.

Eine mechanisch anspruchslosere Ausführung der taktilen Matrix wird im Bild 7.12 gezeigt. Halbrunde Silikonstäbe werden bei Andruck auf runde Metallelektroden gepreßt, wobei sich durch Vergrößerung der Kontaktfläche der Übergangswiderstand verringert. Vorteil dieser Anordnung ist, daß sich durch Bildung von Spalten und Zeilen ein vereinfachtes Prinzip der Signalauslesung anwenden läßt, das im Bild 7.12d dargestellt ist. Durch Schließen des Transistors *D* wird die zugehörige Zeile aktiviert; die einzelnen Elemente werden über einen Analogmultiplexer *C* abgefragt. Durch den Gegenkopplungszweig *B* und *E* wird eine Linearisierung der Kennlinie vorgenommen.

Taktile Matrizen sind aufgrund ihrer Störanfälligkeit, Verschmutzungsgefahr und der durch den Anbringungsort im Greifer bedingten Übertragungsprobleme nur in geringem Maße industriell eingesetzt worden.

Bei neueren Entwicklungen sind die Abtastelektronik und die A/D-Wandlung direkt in einem vollständig gekapselten Sensor integriert [7.10] [7.11].

7.3. Näherungssensoren

Näherungssensoren reagieren auf die Anwesenheit eines Objekts in einem durch das Wirkprinzip und die Wandlerkonstruktion festgelegten Volumen. Die berührungslose Informationsübertragung vom Objekt zum Meßwandler erfolgt durch Felder, Wellen oder Teilchenströme. Die Meßprinzipien nutzen dabei Unterschiede der physikalischen Eigenschaften zwischen dem Objekt und dem erfaßten Volumenbereich aus. Man unterscheidet Anwesenheitssensoren, die nur das Vorhandensein eines Objekts oder eines bestimmten Objektmerkmals erfassen, und Distanzsensoren, die die Entfernung zwischen dem Sensor und dem Objekt oder einem Teil des Objekts bestimmen.

Für die unterschiedlichen Aufgabenbereiche von Industrierobotern scheint eine grobe Einteilung in die Meßbereiche 0,1 bis 10 mm, 10 und 100 mm bis 1 m sinnvoll. Distanz- und Näherungssensoren für den unteren Meßbereich werden nach bekannten Wirkprinzipien für viele Automatisierungsaufgaben industriell gefertigt.

Für die anderen beiden Meßbereiche sind mit der Entwicklung der Robotertechnik zahlreiche Lösungen für die Bestimmung von Relativpositionen, Konturverläufen und weiteren räumlichen Relationen zwischen Objekten erarbeitet worden, von denen einige nachfolgend behandelt werden. Durch die Kombination der Näherungssensoren mit definierten Roboterbewegungen lassen sich häufig mehrdimensionale Meßgrößen gewinnen.

7.3.1. Anwesenheitssensoren

Für die Anwesenheitsbestimmung metallischer Objekte in Abstandsbereichen bis zu einigen Zentimetern werden induktive Sensoren eingesetzt. Sie arbeiten nach dem Prinzip der Schwingkreisdämpfung oder als offene Differentialtransformatoren und können für

bestimmte Schaltabstände geeicht werden [7.12]. Auch offene Kondensatoren lassen sich als Näherungsinitiator einsetzen; dabei wird die Deformation des elektrischen Feldes beim Eindringen von Körpern mit abweichender Dielektrizitätskonstante und die daraus resultierende Kapazitätsvariation genutzt.

Mit Hilfe solcher Schaltinitiatoren lassen sich die Anwesenheit von Objekten im Greifer und das Erreichen bestimmter Positionswerte signalisieren.

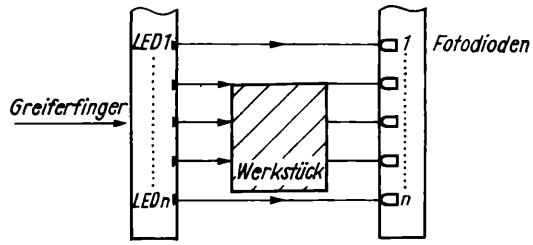


Bild 7.13
LED-Fotodioden-Paare zur Anwesenheitskontrolle und Zentrierung von Objekten im Greifer

Eine von den Materialeigenschaften der Objekte weitgehend unabhängige Anwesenheitsbestimmung ist mit optoelektronischen Lichtschranken realisierbar. Bild 7.13 zeigt die Anordnung von LED-Fotodioden-Paaren in den Greifflächen eines Parallelgreifers. Im statischen Zustand kann mit ihnen neben der Anwesenheitskontrolle eine grobe Information über die Werkstückabmessungen erhalten werden; die erreichbare Genauigkeit hängt dabei von der Anzahl der Lichtschranken ab. Wird der Greifer senkrecht zu den

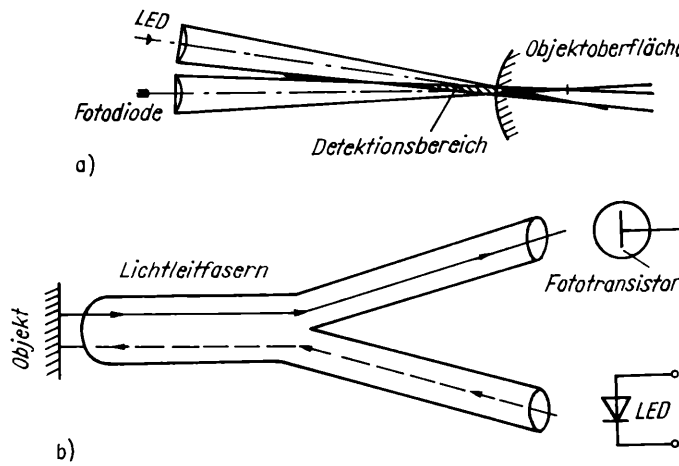


Bild 7.14
a) Reflexlichtschranke; b) Reflexlichtschranke mit Lichtleiterübertragung

Lichtschranken bewegt, läßt sich der Eintritt des Objekts in den Strahl sehr genau bestimmen. Hierdurch sind auch exakte Objektvermessungen oder Lagebestimmungen mit einer oder wenigen Zeilen möglich, wenn mittels des inneren Meßsystems des Industrieroboters der vom Greifer zurückgelegte Weg zwischen Eintritt und Austritt des Werkstücks aus einem Strahl bestimmt werden kann. Damit ist das Werkstück innerhalb des Greifers zentrierbar. Zur Vermeidung der Lichteinstreuung auf benachbarte Fotodioden werden die einzelnen Zeilen zyklisch mit der für die geforderte Genauigkeit notwendigen Taktfrequenz abgefragt.

Fremdlichteinfluß wird durch Schwellwertschalter an jedem Diodenausgang weitgehend unterdrückt.

Neben der Anordnung der Fotodiode im Direktlicht der lichtemittierenden Diode sind auch Reflexlichtschranken zur Anwesenheitskontrolle und für Zentrieraufgaben geeignet

(Bild 7.14a). Dabei werden die Strahlverläufe durch den Abstand LED–Fotodiode und den zwischen den optischen Achsen liegenden Winkel sowie die Linsenbrennweiten so justiert, daß ein definierter Detektionsbereich entsteht.

Bei ausreichender Leistung der Lichtemitterdiode und genügendem Reflexionsvermögen schaltet der Anwesenheitssensor, wenn das Objekt in diesen Bereich eintritt.

Liegen LED und Fotodiode eng beieinander, können die optischen Achsen parallel gelegt werden; der Detektionsbereich fällt dann annähernd mit den Strahlverläufen zusammen. Eine auch für kleine Schaltabstände geeignete Strahlführung erreicht man durch Anwendung von Lichtleitfasern. Das Prinzip ist im Bild 7.14b dargestellt. Der eine Teil des Faserbündels überträgt das emittierte Licht, der andere das reflektierte. Bei streuender Objektoberfläche ist die empfangene Lichtintensität bei Reflexlichtschranken neben dem Reflexionsfaktor von der Entfernung Objekt–Fotodiode abhängig.

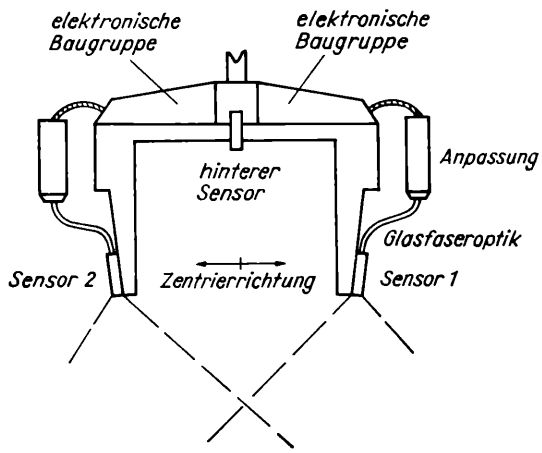


Bild 7.15
Greiferzentrierung mittels zweier Reflexlichtschranken und Intensitätsauswertung

Da die Reflexionseigenschaften am Objekt aber in der Regel stark variieren, z. B. durch Krümmung, Oberflächenstruktur, ist eine direkte Ausnutzung der Amplitudeninformation nicht möglich. Eine Ausnahme bilden Methoden des Intensitätsvergleichs. Im Bild 7.15 sind zwei Anwesenheitssensoren so an den Greiferbacken angeordnet, daß durch die Auswertung der von einem Objekt gestreuten Lichtintensitäten eine Zentrierung des Greifers in der eingezeichneten Bewegungsrichtung erreicht werden kann. Beide Sensoren arbeiten als Reflexlichtschranke mit relativ großem Öffnungswinkel.

Herrschen auf dem Objekt in beiden Strahlkegeln annähernd gleiche Reflexionsverhältnisse, ist die Zentrierung erreicht, wenn bei beiden Sensoren gleiche Ausgangssignalamplituden auftreten.

Für optisch spiegelnde Objekte sind Reflexlichtschranken ungeeignet, wenn der Spiegelwinkel nicht eingehalten werden kann.

7.3.2. Distanzsensoren

Für die Steuerung von Industrierobotern lassen sich drei Arten der Distanzmessung und deren Kombinationen nutzen:

- Messung des Abstands zwischen Meßwandler und einem Punkt der Objektoberfläche,
- Messung der mittleren Entfernung zwischen Meßwandler und Objekt (globale Entfernungsmessung),
- Messung von Distanzen zwischen definierten Objekten oder Teilen dieser Objekte.

Bei metallischen oder ferromagnetischen Werkstoffen läßt sich zur groben Abstandsbestimmung im Nahbereich die Amplitudeninformation induktiver oder kapazitiver Meßwandler auswerten. Ist eine Referenzbahn vorgegeben, kann durch einen Abstandssensor eine Bahnkorrektur, durch zwei Sensoren eine Bahn- und Orientierungskorrektur des Effektors in einer Koordinate vorgenommen werden. Das Prinzip ist im Bild 7.16

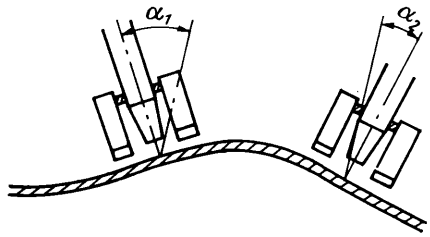


Bild 7.16
Konturverfolgung mit zwei induktiven Distanzsensoren

dargestellt. Induktive Näherungssensoren haben insbesondere bei der Verfolgung von Schweißnähten Anwendung gefunden, wobei die konkrete Sensoranordnung an die Art der Schweißnaht und die erwarteten Abweichungen angepaßt ist [7.13]. Bei der Entwicklung von Sensoren zur Nahtverfolgung beim Schweißen sind hohe Forderungen an die Temperaturfestigkeit, Störsicherheit (Lichtbogen) und die Unempfindlichkeit gegen Verschmutzungen zu erfüllen. Hinzu kommen Forderungen an die Baugröße, um den Arbeitsraum nicht zu stark einzuschränken (Schweißen in Ecken).

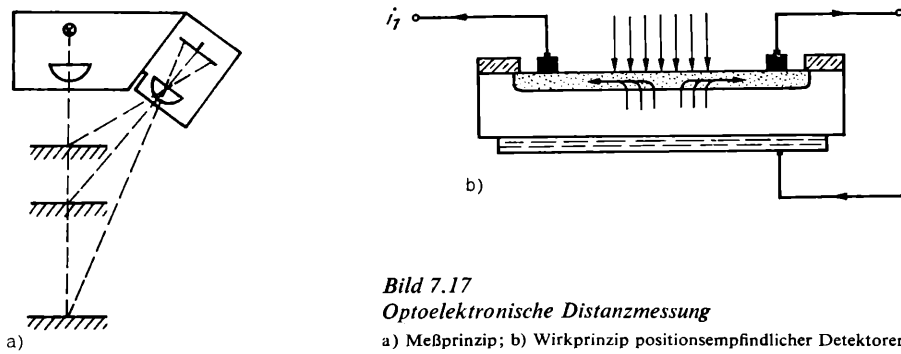


Bild 7.17
Optoelektronische Distanzmessung
a) Meßprinzip; b) Wirkprinzip positionsempfindlicher Detektoren

Für den Bereich mittlerer Entfernungen sind zur Messung von Distanzen zwischen Meßwandler und Objektoberfläche optoelektronische Meßverfahren geeignet, die auf dem Prinzip der Triangulation beruhen. Die Wirkungsweise ist im Bild 7.17 dargestellt. Ein Parallellichtstrahl mit möglichst geringem Durchmesser erzeugt auf der Objektoberfläche einen Lichtfleck; das von der Oberfläche gestreute Licht wird in einem definierten Winkel zur optischen Achse des Strahls mittels einer Linse auf ein optoelektronisches Wandlerelement abgebildet, das die Lage des Lichtpunktes anzeigt. Als Wandlerelemente können in einer Reihe angeordnete Fotodioden oder als Kompaktelement ausgeführte Diodenzeilen dienen (s. Abschn. 7.4.). Der Zusammenhang zwischen Abbildungsort und Entfernung ist nicht linear; beschränkt man sich auf einen kleinen Meßbereich, kann bei entsprechender Wahl der Anordnung und des Abbildungsmaßstabs ausreichende Linearität erreicht werden.

Als analogmessende Wandlerelemente sind positionsempfindliche Streifenfotoempfänger nutzbar, deren Wirkprinzip im Bild 7.17b dargestellt ist. Eine Siliziumfotodiode ist als Streifen (3 bis 4 cm Länge) mit drei Kontakten ausgeführt. Bei Lichteinfall mit

begrenzter Ausdehnung bildet sich ein Fotostrom aus, der sich aufgrund des Substratwiderstands lageabhängig auf die beiden Elektroden verteilt. Diese Aufteilung ist weitgehend unabhängig vom Durchmesser des Lichtflecks, da die Position des Intensitätsschwerpunkts des Lichtbündels gemessen wird. Mit solchen Elementen sind Ortsauflösungen bis zu 5 μm bei hoher Linearität auf dem gesamten Diodenstreifen erreichbar. Die Lagemessung erfolgt durch Auswertung der Fotostromdifferenz. Da diese von der Lichtintensität abhängig ist, sind zur strahlungsflußunabhängigen Lageanzeige spezielle Kompensationsschaltungen erforderlich [7.14].

Ein weiteres Prinzip der optischen Distanzmessung beruht auf der Auswertung der Laufzeit des Lichtes. Dabei werden gemäß Bild 7.18 zwei lichtemittierende Dioden quadraturmoduliert; der Fototransistor empfängt über eine geeignete Linse (im Bild nicht dargestellt) das von einem kleinen Bereich der Objektoberfläche gestreute Licht der beiden LED. Aus der Phasenverschiebung läßt sich durch Vergleich mit den gesendeten Signalen in einem Phasendetektor nach der Beziehung

$$\psi = \arctan\,[(a^2 + x^2)/(b^2 + x^2)]^{3/2} \tag{7.5}$$

die Distanz berechnen. Ein Vorteil dieses Verfahrens ist die Unabhängigkeit von der reflektierten Amplitude. Die Reflexionseigenschaften des Werkstücks bestimmen lediglich die maximal meßbare Distanz, da die Amplitude des vom Objekt gestreuten Lichtes mit dem Quadrat des Abstands abnimmt. Die Modulationsfrequenzen liegen zur Erzielung ausreichender Phasenverschiebungen in der Größe von 1 MHz; dies erfordert den Einsatz schnellschaltender Fotoelemente (GaAs-LED) und analoger Bauelemente mit hoher Grenzfrequenz. Mit dem in [7.2] vorgestellten Distanzsensor können Entfernungen von 10 bis 60 mm bei einer Auflösung von ±1 mm gemessen werden.

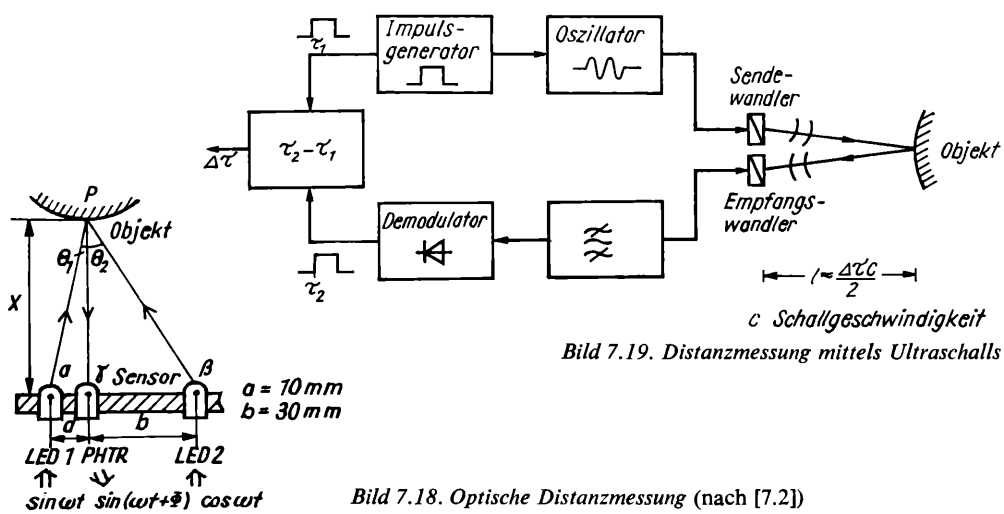


Bild 7.19. Distanzmessung mittels Ultraschalls

Bild 7.18. Optische Distanzmessung (nach [7.2])

Ebenfalls auf der Messung von Laufzeiten beruht die Distanzmessung mittels Ultraschalls. Gemäß Bild 7.19 wird die Laufzeit eines ausgesendeten Schallimpulses, der am Objekt reflektiert wird, direkt ausgewertet. Mit solchen Anordnungen lassen sich aufgrund der geringen Richtwirkung von Ultraschallwandlern, die heute vorwiegend aus piezokeramischem Material bestehen, ohne komplizierte Verfahren der Signalauswertung nur minimale oder mittlere Entfernungen zwischen Objekt und Wandler bestimmen.

Die Messung kleinster Abstände oder Positionsdifferenzen spielt insbesondere bei Montageaufgaben eine Rolle. Da die Positioniergenauigkeit von Industrierobotern für die Präzisionsmontage in der Regel nicht ausreicht, werden spezielle Nachführeinrichtungen eingesetzt, die die Positionsdifferenzen zwischen den zu fügenden Teilen ausgleichen. Im Bild 7.20 ist eine Lösung dargestellt, bei der die Positionsdifferenzen mittels pneumatischer Sensoren erfaßt werden. Das Prinzip wird im Bild 7.20a gezeigt. Die Druckluft strömt aus je zwei Öffnungen (*2*) aus, die durch die Montageöffnung (*3*) unterschiedlich abgedeckt werden. Die Staudrucke steuern eine pneumatische Brücke; bei gleicher Abdeckung ist die Druckdifferenz Null. Bild 7.20b zeigt einen vollständigen Fügemechanismus für rotationssymmetrische Teile. Das zu fügende Teil (*1*) befindet sich zwischen den mit zwei Sensorpaaren ausgestatteten Greiferbacken; die Montageöffnung (*2*) besitzt eine

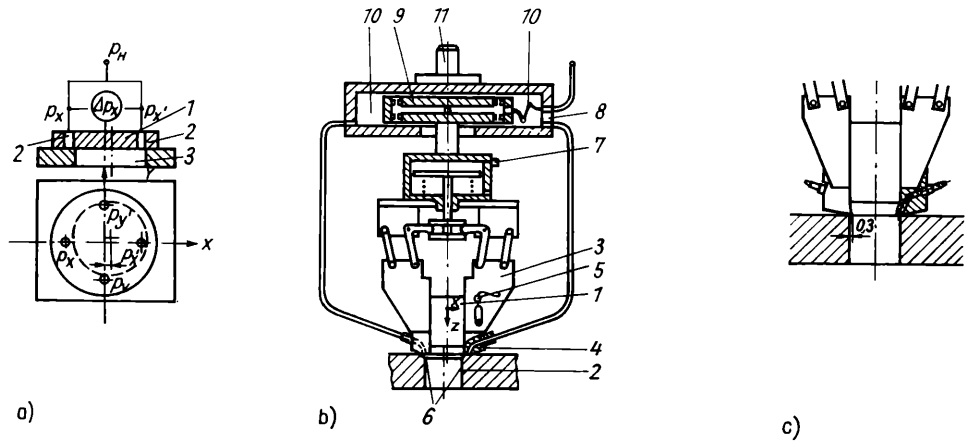


Bild 7.20. Pneumatischer Fügemechanismus

a) Meßprinzip; b) Zentrierung rotationssymmetrischer Teile über Bohrungen mit Fase; c) Zentriermechanismus für Bohrungen ohne Fase

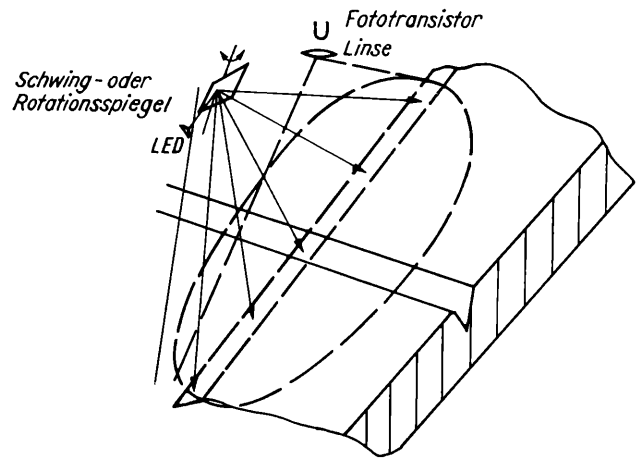


Bild 7.21
Prinzip der Lagemessung durch Auswertung der Reflexionsamplitude

Fase. Der Differenzdruck steuert in den Druckkammern (*10*) einen Schieber, der den Greifer parallel zur Objektoberfläche bewegt, bis die Druckdifferenz Null ist. Der Greifer selbst wird über den Anschluß (*7*) ebenfalls pneumatisch betätigt. Im Bild 7.20c ist eine Anordnung dargestellt, mit der auch Teile ohne Fase justiert werden können [7.7]. Ein Vorteil solcher pneumatischer Sensoren besteht in ihrer Robustheit und Störunanfällig-

keit; im Wirkungsbereich der Sensoren befindliche Schmutzpartikel werden durch den Luftstrom entfernt. Ein Nachteil ist, daß sie für unterschiedliche Teilegrößen und Fügeaufgaben speziell eingerichtet oder konstruiert werden müssen. Wegen der großen Zahl gleichartiger oder ähnlicher Montageaufgaben sind solche Fügemechanismen trotzdem sehr effektiv nutzbar.

Mit Hilfe der sog. Lichtschnittechnik können relative Entfernungen zwischen markanten Teilen von Objekten vermessen werden. Dabei sind sowohl Amplituden- als auch Winkelinformationen auswertbar. Im Bild 7.21 ist ein Verfahren zur Schweißnahtverfolgung dargestellt, bei dem das unterschiedliche Reflexionsvermögen bestimmter Werkstückbereiche ausgenutzt wird. Durch einen beweglichen Spiegel wird ein Lichtpunkt auf einer Linie über das Objekt bewegt; die momentane Intensität des reflektierten Lichtes wird mittels eines ungerichtet empfangenden Fototransistors ausgewertet und bildet mit der momentanen Stellung des Spiegels einen zweidimensionalen Meßwert. Ist die Distanz zwischen Sensorelementen und Objektoberfläche bekannt, können Abstände auf dem Objekt bestimmt werden. Nachteilig wirkt sich dabei besonders die Abhängigkeit von den Reflexionseigenschaften aus; durch Verschmutzungen und unterschiedliche Reflexionsfaktoren bei gleichem Oberflächenverlauf kommt es leicht zu Fehlinformationen.

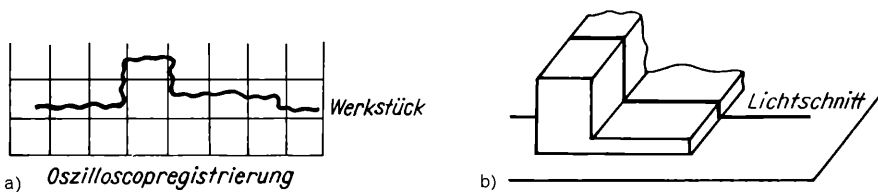
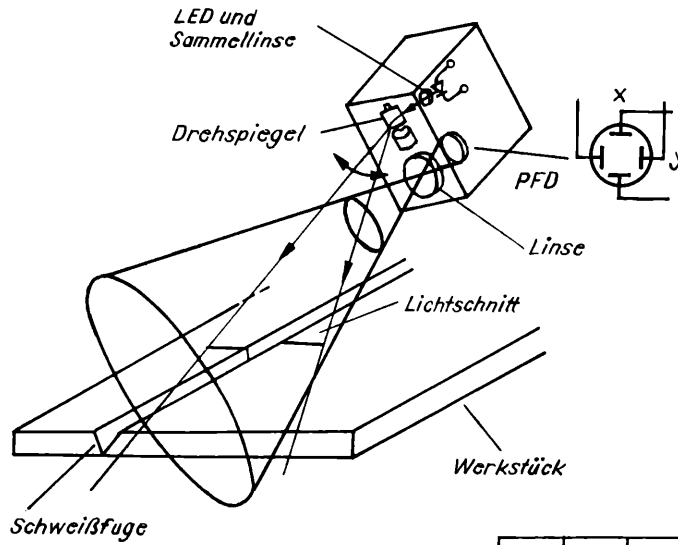


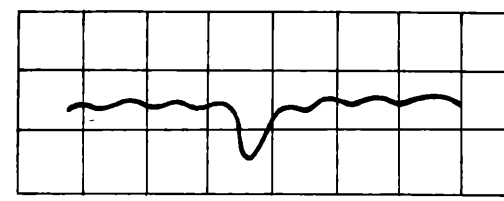
Bild 7.22. Lichtschnittechnik
a) Prinzip; b) oszillografiertes Meßsignal

Wird der vom bewegten Spiegel erzeugte Lichtschnitt in einem Winkel zur Ebene des Strahlverlaufs betrachtet, lassen sich die Objektkonturen direkt zweidimensional abbilden (Bild 7.22). Ein nach diesem Prinzip arbeitender Sensor zur Schweißnahtverfolgung wird in [7.15] beschrieben. Die geometrische Anordnung wird im Bild 7.23a gezeigt. Der Lichtschnitt wird durch eine im Infrarot arbeitende Leistungs-LED und einen Drehspiegel erzeugt. Der Lichtpunkt auf dem Werkstück wird über eine Sammellinse auf eine positionsempfindliche Flächendiode abgebildet. Diese arbeitet wie die im Bild 7.17 dargestellte Streifendiode – nur mit dem Unterschied, daß mittels zweier Kontaktpaare die Lichtfleckposition in zwei Dimensionen aus den Fotostromdifferenzen gemessen werden kann. Bild 7.23b zeigt den Positionsverlauf des abgebildeten Lichtflecks für eine Schweißfuge. Neben der Ermittlung des Nahtverlaufs ist aus der senkrechten Verschiebung des Lichtschnitts auf der positionsempfindlichen Diode auch die Höhe des Sensors über dem Werkstück berechenbar. Die Signalverarbeitung wird im Bild 7.23c gezeigt. Die LED wird zur Unterdrückung von Störlicht mit 10 kHz pulsmoduliert; der Drehspiegel wird mit 100 Hz synchronisiert. Der Positionsverlauf wird auf einem Mikrorechner ausgewertet. Hierzu wird durch Einstellen einer Sollposition an einem Werkstück mit Vorgabe der Schweißnaht (Stumpfnaht, V-Naht, Kehlnaht) aus der Abweichung zu den gemessenen Werten direkt die Korrektur für die Bahnkurvenwerte berechnet.

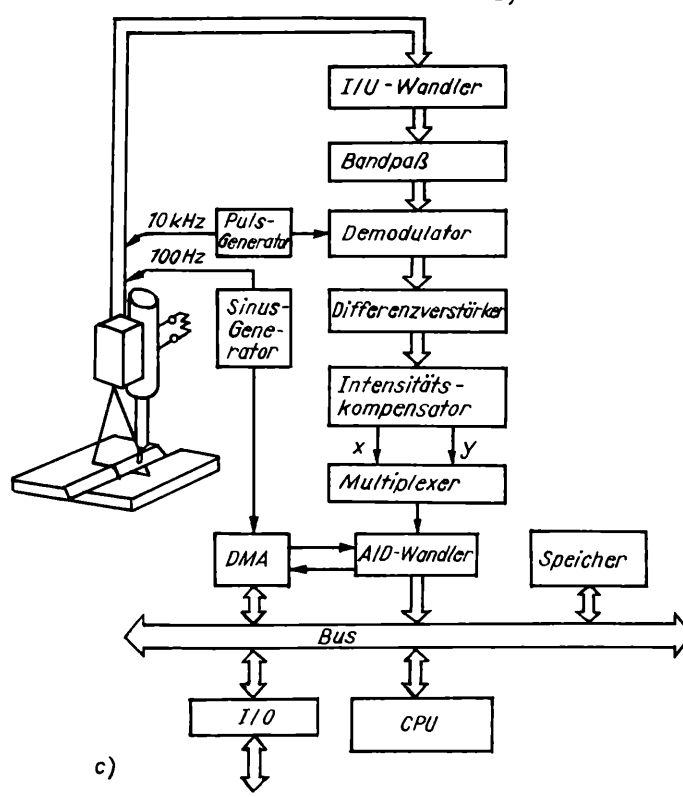
Die Vermessung von Objekten ist auch mit Distanzmeßsystemen möglich, wenn anstelle der Abtastung mit Dreh- oder Schwingspiegeln eine Scanierung durch die Bewegung des am Greifer befindlichen Sensors in einer definierten Weise erfolgt. Die einfachste Möglichkeit hierzu ist die Bewegung von Manipulatorgelenken. Die Distanzwerte lassen sich dabei direkt den Gelenkkoordinaten zuordnen.



a)



b)



c)

Bild 7.23
Schweißnahterkennung mit optischem Lagemeßsystem (nach [7.15])

a) Sensorsystem
b) Signalverarbeitung

7.4. Visuelle Erkennungssysteme

Die visuelle Erkennung hat aufgrund der Komplexität der erfaßbaren Umweltinformation die größte Bedeutung für die Erhöhung der Adaptionsfähigkeit und Flexibilität von Robotersystemen. Durch die geringe Wellenlänge des Lichtes und der Unabhängigkeit der Ausbreitungsgeschwindigkeit von Umwelteinflüssen ist potentiell ein hohes Auflösungsvermögen vorhanden.

In Industrierobotersystemen werden visuelle Erkennungssysteme für folgende Aufgaben eingesetzt:

- lagerichtiges Greifen undefiniert abgelegter oder bewegter Werkstücke,
- Sortieren unterschiedlicher Werkstücke und Erkennung von Werkstückfehlern,
- Naht- und Kantenverfolgung bei technologischen Operationen (z. B. Schweißen),
- Formerkennung (z. B. beim Farbspritzen),
- Erkennung von relativen Lagen bei Montageoperationen,
- Qualitätsüberwachung.

Die Bildverarbeitungsmethoden lassen sich auf drei grundlegende Aufgabenklassen zurückführen, die einzeln oder in Kombination zur Anwendung kommen:

1. Erkennen von Werkstücken anhand vorgegebener Abbildungsmuster,
2. Bestimmung der Lage und der Orientierung von Handhabungsobjekten in bezug auf ein vorgegebenes Koordinatensystem,
3. Erkennen und Lokalisierung einzelner Werkstückstrukturen (z. B. Löcher, Ausbrüche, Grat, Konturen).

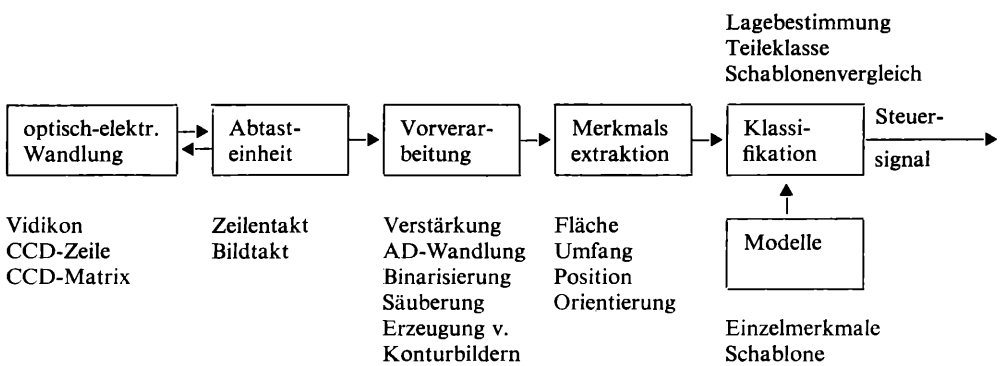


Bild 7.24. Prinzipieller Aufbau visueller Erkennungssysteme und typische Verarbeitungsalgorithmen

Man unterscheidet zwischen 2dimensionaler und 3dimensionaler Bilderkennung.

Der prinzipielle Aufbau visueller Erkennungssysteme ist im Bild 7.24 dargestellt. Die *optisch-elektrische Wandlung* beruht auf der Freisetzung elektrischer Ladungen in einem geeigneten Material und erzeugt aus dem optischen Abbild ein entsprechend der Geometrie des Wandlers gerastertes Ladungsbild. Dieses wird durch die *Abtasteinheit* in ein amplitudenmoduliertes, zeitdiskretes Signal umgeformt. Da die weitere Verarbeitung der Bildinformation ausschließlich digital erfolgt, wird in der *Vorverarbeitungsstufe* zunächst eine A/D-Wandlung vorgenommen; weitere Vorverarbeitungsalgorithmen richten sich nach der Art der Merkmale, die aus dem Bild gewonnen werden sollen. Durch Bildreinigungsalgorithmen, Hervorhebung von Konturen und Binarisierung wird die Bildinformation so weit reduziert, daß nur die für die Aufgabe relevanten Teile enthalten sind.

Im nachfolgenden *Merkmalsextraktor* werden aggregierende Beschreibungsmittel gebildet, die morphometrischen (z. B. Fläche der Silhouette, Durchmesser, Flächenschwerpunkt usw.), textuellen (z. B. „über", „größer", „Loch im Teil vorhanden" usw.) oder auch spektralen Charakter haben. Der *Klassifikator* vergleicht diese Merkmale mit den gespeicherten Solldaten und generiert ein geeignetes Steuersignal für den Manipulator.

Die Modellmerkmale können programmiert oder in einer Lernphase durch „Vorzeigen" eines entsprechenden Teiles des Sortiments ermittelt werden. Zusammen mit den zugelassenen Toleranzen definiert der so eingegebene Merkmalsvektor die Objektklasse, für n Objekte sind ebenso viele Objektklassen erforderlich. Fällt der im Erkennungsprozeß ermittelte Merkmalsvektor nicht in eine der zugelassenen Objektklassen, ist durch den Klassifikator ein Rückweisungssignal zu erzeugen; durch Aufruf eines entsprechenden Bewegungsprogramms kann dieses Teil dann vom Industrieroboter ausgesondert werden. Gemäß den drei Aufgabenklassen ergeben sich folgende funktionelle Schnittstellen zur Industrierobotersteuerung:

- Erkennung – Aufruf eines Steuerprogramms aus dem Anwenderspeicher,
- Positions- und Orientierungserkennung – Übergabe von Sollpositionen an die Bewegungssteuerung,
- Lokalisieren von Werkstückstrukturen – Übergabe von Bahnpunkten.

Die Erkennung einzelner Werkstückstrukturen wird daneben auch zur Qualitätssicherung eingesetzt, z. B. zur Erkennung und Bewertung von Löchern, Grat, Ausbrüchen.

Die Steuerung von Industrierobotern nach visuell ermittelten Daten erfordert in den meisten Fällen kurze Verarbeitungszeiten der Erkennungsalgorithmen, um die Zeitbedingungen des technologischen Zyklus einzuhalten. Neben der Auswahl schneller Algorithmen und einfacher Merkmalsextraktionsmethoden werden häufig effektive Hardwareimplementierungen vorgenommen. In der Regel werden durch Einschränkungen der Objektlagen und zusätzliche Bedingungen wie

- Teile liegen vereinzelt vor,
- Teile berühren sich nicht,
- Teile nehmen nur eine bestimmte Zahl stabiler Lagen ein, oder
- es kommt nur eine bestimmte Klasse von Teilen vor

der Verarbeitungsaufwand zur Erfüllung von Echtzeitforderungen eingeschränkt. Dies geschieht auf Kosten eines zusätzlichen Aufwands in der Peripherie des Roboters.

In den nachfolgenden Abschnitten werden diese Verarbeitungsschritte und einige den Algorithmen zugrunde liegende Methoden nacheinander am Beispiel der 2-D-Erkennung behandelt.

Verfahren zur 3-D-Erkennung unterscheiden sich hiervon durch das wesentlich größere zu verarbeitende Datenmassiv einschließlich des zu speichernden Modellwissens und andere Methoden der Bildaufnahmetechnik, die im Abschnitt 7.4.6. dargestellt sind.

7.4.1. Bilderzeugung und Bildwandlung

Merkmalsgewinnung und Mustervergleich erfolgen durch Auswertung eines binären 2-D-Bildes des Objekts. Man unterscheidet dabei zwischen einem Silhouettenbild und einem Konturbild (Bild 7.25). Während beim Silhouettenbild die vollständige Schwarzweißinformation vorliegt, werden beim Konturbild nur die Schwarzweißübergänge erfaßt. Der Übergang vom Silhouettenbild zum Konturbild stellt eine Differenzierung dar und bedeutet keinen Informationsverlust.

Ziel der Bilderzeugung und Bildwandlung ist die Erstellung eines elektrischen Abbildsignals, aus dem mit möglichst geringem Verarbeitungsaufwand ein der Silhouette oder Kontur entsprechendes Signal gewonnen werden kann.

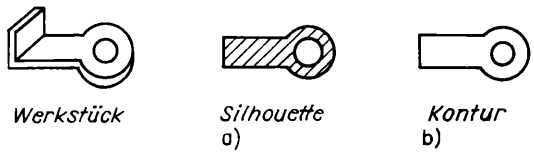


Bild 7.25
Silhouetten- (a) und Konturbild (b)

Beleuchtungstechnik, Bildfenster und Anordnung der Objekte sind bei Beachtung der technologischen Gegebenheiten so zu gestalten, daß bereits die optische Abbildung auf dem Bildwandler kontrastreich ist und nur die zu erkennenden bzw. zu unterscheidenden Objekte enthält. Die günstigsten Voraussetzungen zur Erzeugung eines kontrastreichen Binärbilds bietet die Durchlichtmethode. Das Objekt wird auf einer diffus streuenden, durchscheinenden Unterlage angeordnet, die von unten beleuchtet ist (Bild 7.26a). Da undurchsichtige Schmutzpartikel ebenfalls mit gutem Kontrast abgebildet werden, ist die Durchlichtmethode praktisch nur in wenigen Fällen einsetzbar.

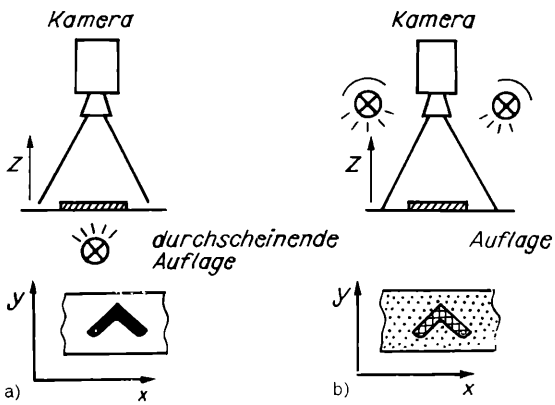


Bild 7.26
Beleuchtungstechniken der 2-D-Erkennung
a) Durchlichtmethode;
b) Auflichtmethode

Die im Bild 7.26b schematisch dargestellte Auflichtmethode beruht auf dem unterschiedlichen Reflexionsvermögen von Objekt und Untergrund.

Weitere Beleuchtungstechniken sind die aus der Mikroskopie bekannten Hellfeld- und Dunkelfeldbeobachtung, bei denen Lichtquelle und Kamera so angeordnet werden, daß entsprechend dem Reflexionsgesetz ebene Flächen hell oder dunkel erscheinen, während an Kanten oder Störungen in der Bildoberfläche eine Streuung auftritt.

Die Wahl der günstigsten Beleuchtung hängt neben den technologischen Möglichkeiten von der Beschaffenheit der Objekte und der durch die Erkennung zu lösenden Aufgabe ab und wird in der Regel durch Vorversuche zu klären sein.

Das von der Kamera erfaßte 2dimensionale Silhouettenbild eines i. allg. 3dimensionalen Werkstücks ist gemäß Bild 7.26 von seiner Ausdehnung in x- und y-Richtung und damit von der momentanen Lage abhängig. Drehungen des Teiles um die x- oder y-Achse ändern außer bei vollständiger Rotationssymmetrie die Silhouette. In diesem Abschnitt wird der Begriff Lage nur auf die Drehlage bezüglich dieser beiden Koordinatenachsen angewendet. Je nach Teilebeschaffenheit lassen sich unterschiedlich viele stabile, instabile und indifferente Teilelagen unterscheiden. Indifferente Gleichgewichtslagen können zu unendlich vielen möglichen Silhouettenbildern führen; eine Erkennung ist dann im allgemeinen Fall nicht möglich.

Die Drehlage des Werkstücks um die z-Achse wird in der visuellen Erkennung als Orientierung bezeichnet. Änderung der Teileorientierung ändert das Silhouettenbild nicht, wenn man von optisch bedingten Abbildungsfehlern absieht. Die x- und y-Koordinaten eines ausgewählten Punktes des Silhouettenbilds bezeichnet man als Position; in der Regel werden zur Positionsangabe die x- und y-Werte des Flächenschwerpunktes der Silhouette genutzt.

Das objekteigene Frame kann somit nur bezüglich der x- und y-Werte des Positionsvektors im Basiskoordinatensystem (s. Abschn. 4.6.2.) und des Orientierungswinkels Φ variieren. Für unterschiedliche Teilelagen sind ebenso viele Objektframes zu definieren, in denen z. B. mögliche Greifpositionen T_g angegeben werden können.

Die Wandlung des mittels einer geeigneten Optik erzeugten Abbilds in ein elektrisches Signal erfolgt bei allen bekannten Bildwandlersystemen durch Freisetzung von Ladungen in einem halbleitenden Material. Im Prinzip besteht jedes Bildwandlungssystem aus einer mehr oder minder großen Zahl von Kondensatoren, die durch fotoelektrische Effekte aufgeladen werden. Die Größe der Ladung hängt dabei von der Wellenlänge und Intensität des Lichtes und der Dauer der Aufladung ab. Durch die Strukturierung der einzelnen Kondensatoren auf der Wandlerfläche entsteht ein sog. Rasterbild, das die mögliche Auflösung begrenzt.

Das älteste und am weitesten verbreitete Bildwandlersystem ist die Fernsehkamera. Über ein Ablenksystem tastet ein Elektronenstrahl zeilenweise die Bildpunkte ab und entlädt die Kondensatoren; der dabei auftretende Entladungsstrom entspricht der Helligkeit des Bildpunktes. Bei üblichen Fernsehnormen wird das Bild in 625 Zeilen zerlegt, wobei das Abtasten jeder Zeile 64 µs dauert, ein vollständiges Bild wird in $\frac{1}{25}$ s gewandelt. Die Positionsbestimmung eines Bildpunktes kann bei gegebener Bild- und Zeilenfrequenz unmittelbar über eine Zeitmessung erfolgen.

Aufgrund der durch die Nichtlinearität der horizontalen und vertikalen Elektronenstrahlablenkung entstehenden Abbildungsfehler sind Fernsehkameras für industrielle Erkennungs- und Meßaufgaben nur bedingt einsetzbar.

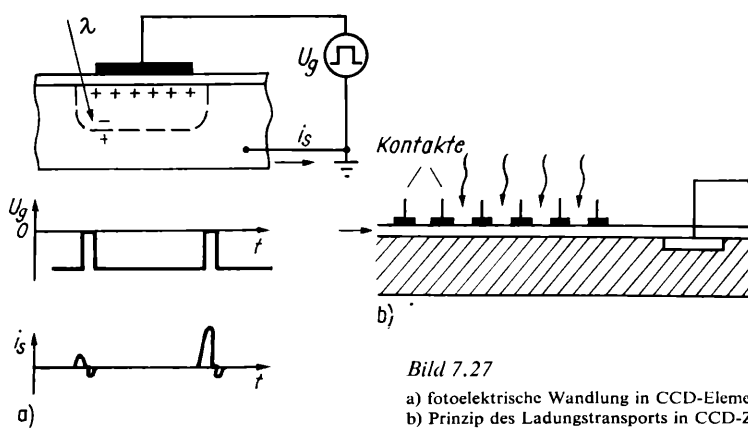


Bild 7.27
a) fotoelektrische Wandlung in CCD-Elementen
b) Prinzip des Ladungstransports in CCD-Zeilen

Exakte Bildwandlungen ermöglichen CCD-Kameras (*c*harge *c*oupled *d*evices), mit deren Entwicklung 1970 begonnen wurde.

Das Prinzip der fotoelektrischen Wandlung in CCD-Elementen wird im Bild 7.27a gezeigt. Auf einem p-dotierten Siliziumsubstrat ist eine 100 bis 300 nm dünne Siliziumdioxidschicht aufgebracht. Durch Anlegen einer Spannung zwischen Metallelektrode und Substrat bildet sich um die Elektrode eine Verarmungszone, die durch fotoelektrisch

erzeugte Elektronen aufgefüllt wird. Setzt man die Spannung kurzzeitig Null, bildet sich ein Ladestrom, der dem Photonenstrom proportional ist. Ordnet man gemäß Bild 7.27b mehrere solcher Kondensatoren in einer Reihe an, kann die Ladung eines Kondensators durch Anlegen geeigneter Taktspannungen an den Nachbarelektroden von diesen übernommen und an der letzten Elektrode seriell ausgelesen werden. Dabei sind zum definierten Ladungstransport drei Elektroden notwendig – eine gibt das Ladungspaket ab; eine andere übernimmt die Ladung, und eine weitere verhindert das Zurückfließen. Nach diesem Prinzip lassen sich CCD-Zeilen aufbauen mit Bildpunktabmessungen bis zu 10 μm × 10 μm. Da während des Ladungstransports durch die einzelnen Elemente zusätzliche Ladungen durch Lichteinfall entstehen, muß die gesamte Zeile während sehr kurzer Zeit ausgelesen werden. Deshalb wird dieses Ausleseprinzip heute kaum noch angewendet. Eine zweite Möglichkeit zum Auslesen der Bildpunkte wird im Bild 7.28 gezeigt. Die CCD-Elemente – hier als Ersatzschaltbild durch Diode und Kondensator dargestellt – werden über einen MOS-Transistor nacheinander auf die gemeinsame Ausgangsleitung geschaltet. Der hierzu erforderliche Taktimpuls wird durch ein ebenfalls auf dem Substrat integriertes Schieberegister erzeugt. Dabei wird der Kondensator auf 5 V

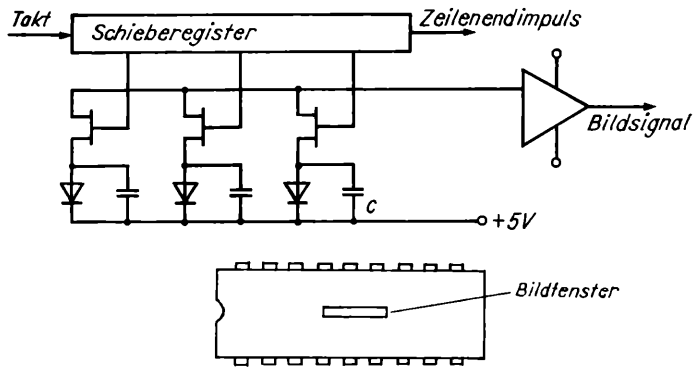


Bild 7.28
Ersatzschaltbild einer CCD-Zeile, das Auslesen erfolgt mittels getakteten Schieberegisters auf die gemeinsame Videoleitung

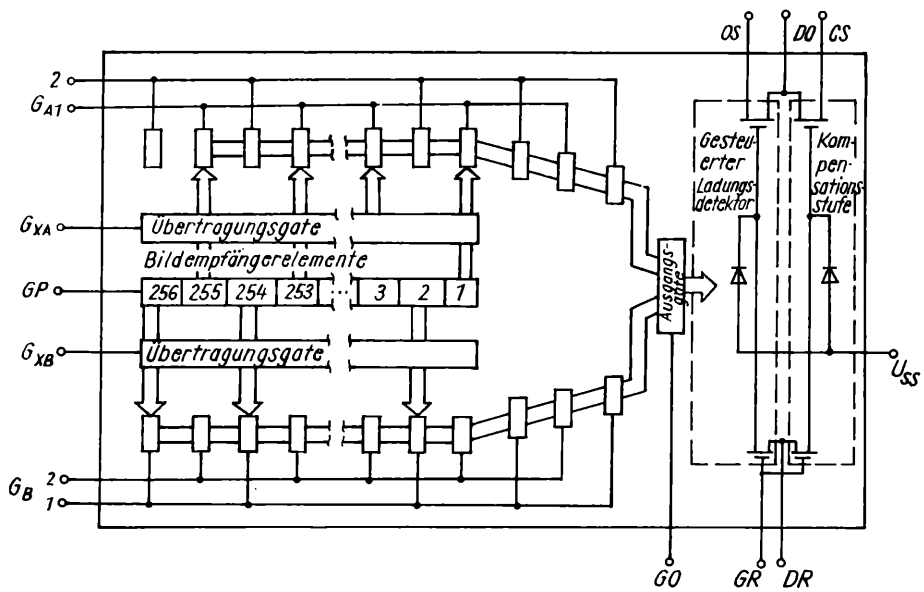


Bild 7.29. Aufbau der CCD-Zeile L110 des VEB Werk für Fernsehelektronik [7.17]

aufgeladen; bis zum nächsten Taktimpuls findet eine dem Fotostrom der Diode proportionale Entladung statt. Der auf der Ausgangsleitung entstehende Fotostrom ist der Beleuchtung proportional. Da die fotoelektrische Ladungsbildung der Beleuchtungszeit proportional ist, kann durch Veränderung der Taktzeiten das Ausgangssignal geregelt werden. Hierdurch lassen sich Übersteuerungen (vollständige Kondensatorentladung) vermeiden und die Empfindlichkeit der Kamera an die Beleuchtungsstärke anpassen. Am Ende einer Ausleseperiode wird vom Schieberegister ein längerer Impuls erzeugt, der das Zeilenende markiert.

Ein weiteres Ausleseprinzip benutzt zwei nebeneinanderliegende CCD-Zeilen. In der ersten findet die fotoelektrische Wandlung statt, während die zweite Zeile gemäß Bild 7.27b den Ladungstransport übernimmt und nicht beleuchtet wird. Den Aufbau einer nach diesem Prinzip arbeitenden, handelsüblichen Zeile zeigt Bild 7.29. Die 256 lichtempfindlichen CCD-Elemente sind in einer Reihe angeordnet und durch einen mäanderförmig eindiffundierten Trennkanal voneinander separiert. Auf jeder Seite der Bildempfängerzeile ist ein Übertragungsgate angeordnet, das im geöffneten Zustand die Ladung jedes zweiten Bildelements in eine Transport-CCD-Zeile übergibt. Die Ladungen der ungeradzahlig bezifferten Elemente gelangen in die Transportzeile A, die der geradzahligen in die Transportzeile B. Diese Ladungsübernahme erfolgt in der Transportpause in den Zeilen A und B. Während der nachfolgenden Lichtintegration in den Bildelementen sind die Übertragungsgates gesperrt, und es erfolgt das Auslesen der Transportzeilen durch Taktimpulse zwischen den Steuerleitungen G. Die Anordnung der beiden letzten Elemente der Analogregister A und B bewirkt eine alternierende Einspeisung der Ladungsbilder über ein Ausgangsgate in den Ladungsdetektor, so daß die richtige Reihenfolge der Ladungsbilder gewährleistet ist.

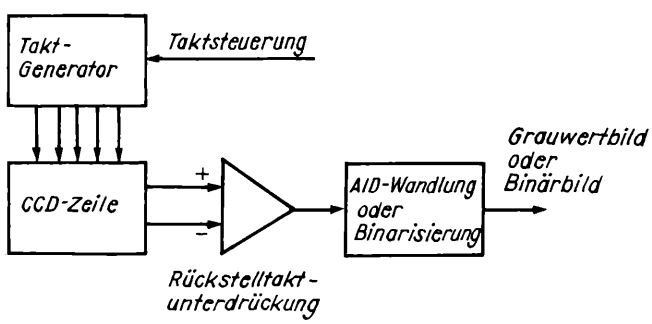


Bild 7.30
Beschaltung einer CCD-Zeile

Die Transportregister übergeben die Ladungsbilder an eine vorgespannte Diode, die mit einer Potentialänderung reagiert. Diese Potentialänderung steuert die Gateelektrode eines MOS-Ausgangstransistors, der eine Videospannung an SO erzeugt. Der Rückstelltransistor T_r wird durch einen Rückstelltaktimpuls an GR gesteuert und sichert das Nachladen der Detektordiode zur Wiederherstellung der Vorspannung, bevor ein neues Ladungsbild eintrifft. Die Kompensationsstufe wird nur vom Rückstelltaktimpuls gesteuert und liefert an CS ein Signal, das gleich den im Videosignal enthaltenen Rückstelltaktimpulsen ist. Dieses Signal wird in einem externen Differenzverstärker zur Unterdrückung des Rückstellimpulses im Videosignal benutzt.

Die prinzipielle Beschaltung einer CCD-Zeile wird im Bild 7.30 gezeigt. Im Taktgenerator werden alle erforderlichen Steuersignale erzeugt; die Taktsteuerung dient der Anpassung der Ladungsintegrationszeit an die Beleuchtungsverhältnisse. Bei gutem Kontrast kann das Videosignal mittels eines Schwellwertschalters direkt in ein Binärbild ge-

wandelt werden; andernfalls ist eine A/D-Konvertierung notwendig, bevor durch weitere Verarbeitungsschritte, die im folgenden Abschnitt besprochen werden, eine Binarisierung durchgeführt wird. Da mit CCD-Zeilen nur ein linienhafter Bildausschnitt erfaßt wird, ist zur Abbildung der vollständigen Silhouette eine Relativbewegung von Kamera und Objekt senkrecht zur abgebildeten Linie in der Bildebene erforderlich. Dies kann entweder durch Bewegung der Kamera oder des Werkstücks (Transportband, lineare Verfahreinheit) erfolgen. Die Transportgeschwindigkeit ist dabei so zu steuern, daß die nacheinander abgetasteten Zeilen so dicht beieinander liegen wie die Bildpunktelemente, damit eine gleichmäßige Auflösung des Bildes erreicht wird.

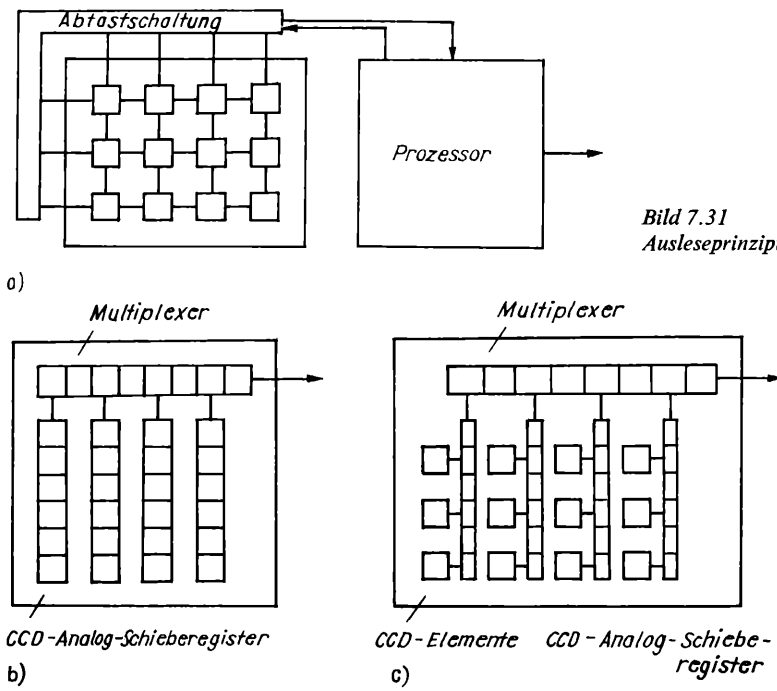


Bild 7.31
Ausleseprinzipien bei CCD-Matrizen

Die Anzahl der Bildpunktelemente einer Zeile n_y und die Zahl der beim Transport abgetasteten Zeilen n_x bestimmen die Anzahl der Rasterpunkte des Bildes und damit die erreichbare Auflösung. Ordnet man mehrere CCD-Zeilen nebeneinander auf einem Chip an, wird während eines Abbildungsvorgangs die vollständige 2-D-Information erhalten. Im Bild 7.31 sind die möglichen Ausleseprinzipien für CCD-Matrizen dargestellt. Am Ausgang wird ein Videosignal gebildet, in dem alle Zeilensignale aufeinander folgen. Zwischen den Zeilen werden Zeilenendsignale eingefügt; zwischen den vollständigen Bildsignalen liegen Bildendimpulse. CCD-Matrizen werden bereits mit bis zu 800 × 800 Bildpunkten gefertigt.

Werden nur geringe Forderungen an das erreichbare Auflösungsvermögen der Kamera gestellt, können durch matrixförmige Anordnungen von einzelnen Fotoelementen preisgünstige Kameras hergestellt werden. Solche Sensoren sind besonders für Überwachungs- und Kontrollfunktionen in unmittelbarer Objektnähe, z. B. am Greifer eines IR, einsetzbar; das Ausleseprinzip entspricht dabei dem im Bild 7.31 c dargestellten. Ist die Anbringung der Kamera aus Platzgründen am Beobachtungsort direkt nicht möglich, kann zur Übertragung der optischen Bilder auch die Lichtleittechnik eingesetzt werden. Auf die im Prinzip aus der Endoskopie bekannten Techniken soll hier nicht weiter eingegangen werden.

7.4.2. Bildvorverarbeitung

Die erforderlichen Algorithmen und Verarbeitungsschritte zur Bildsäuberung und -restaurierung ergeben sich aus den Eigenschaften des von der Kamera erfaßten Silhouettenbilds und den anzuwendenden Merkmalsextraktionsverfahren. Man unterscheidet Algorithmen zur Bildglättung, -säuberung, -restaurierung, Konturenverschärfung und Kantendetektion. Ziel ist die Herstellung eines binären Silhouettenbilds des zu erkennenden Objekts bei Unterdrückung aller Störungen – unabhängig davon, ob diese im Objekt-

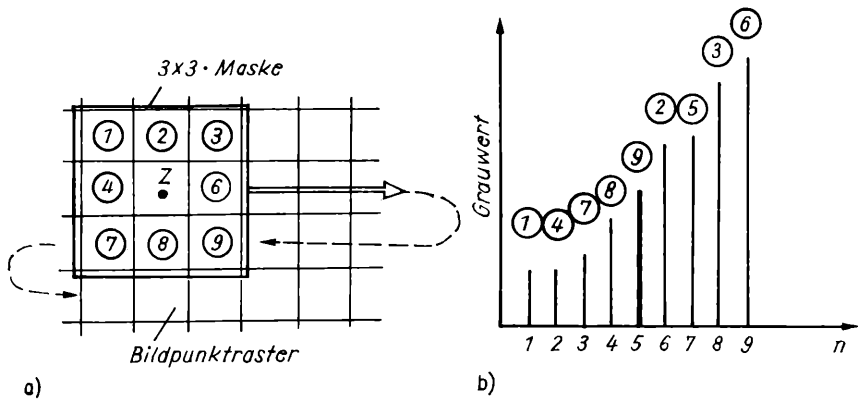


Bild 7.32. Wirkungsweise des Medianoperators

a) 3 × 3-Maske und deren Verschiebung im Bildpunktraster; b) Histogramm, beim Medianoperator wird der Grauwert mit $n = 5$ im Histogramm dem Zentralpunkt Z als neuer Grauwert zugewiesen

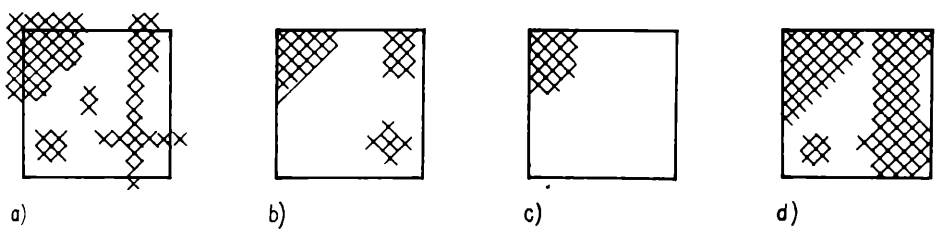


Bild 7.33. Anwendung von Rangordnungsoperatoren auf ein Binärbild

a) Original; b) $R_{3,5}$ (Medianoperator); c) $R_{3,3}$; d) $R_{3,7}$

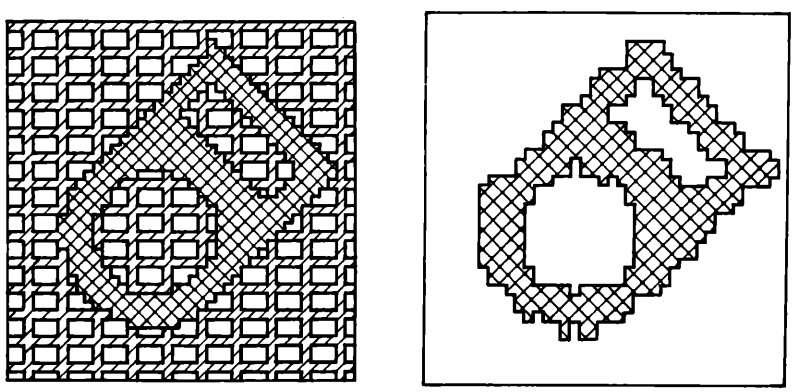


Bild 7.34. Anwendung eines Rangordnungsoperators $R_{3,3}$ zur Unterdrückung einer Gitterstruktur

bereich (z.B. durch Verschmutzung der Unterlage) oder während der Aufnahme und Verarbeitung entstanden sind.

Liegt ein kontrastreiches Bildmaterial vor, läßt sich aus diesem durch Einführen einer Schwelle relativ einfach ein Binärbild erzeugen. Zur Vorverarbeitung eines gerasterten Grauwertbilds ist zunächst eine A/D-Umsetzung notwendig; je nach Kompliziertheitsgrad des Bildes werden 8, 16 oder 32 Graustufen unterschieden.

Charakteristisch für Vorverarbeitungsalgorithmen sind lokale Bildverarbeitungsverfahren, sog. Maskenoperationen. Sie erfüllen gleichzeitig mehrere Aufgaben. Das Prinzip beruht darauf, daß die ursprünglichen Grauwerte oder Binärwerte eines Rasterpunktes durch arithmetische oder logische Verknüpfung mit den Umgebungspunkten neu bestimmt werden. Die Größe solcher Masken kann beliebig gewählt werden; aus Aufwandsgründen kommen vorzugsweise 3×3-Masken zur Anwendung. Die Wirkungsweise soll am Beispiel des Medianoperators erläutert werden, der von *Pratt* [7.19] erstmals in der Bildverarbeitung angewendet wurde. Das Prinzip des Medianoperators besteht darin, daß die Grauwerte aller Maskenpunkte der Größe nach in einem Histogramm geordnet werden und der in der Mitte liegende Wert dieser Folge (bei einer 3×3-Maske der Grauwert an der fünften Stelle) dem Zentralpunkt Z als neuer Grauwert zugewiesen wird (Bild 7.32). Dieser Operator wird Bildpunkt für Bildpunkt nacheinander auf jede Zeile angewendet, wobei stets die darüber- und darunterliegende Zeile benötigt werden. Der Medianoperator mit 3×3-Maske bewirkt eine Glättung der Kontur, die Unterdrückung von Störflecken mit einer Ausdehnung bis zu vier Rasterpunkten sowie die Beseitigung von Linien mit der Breite eines Rasterpunktes. Bild 7.33 veranschaulicht diese Eigenschaften für ein Binärbild; die Bildgröße wird durch diese Maskenoperationen an den Außenkanten um eine Rasterbreite verringert. Die Leistungsfähigkeit solcher Maskenoperatoren verdeutlicht Bild 7.34, wo eine Gitterstruktur der Objektauflage vollständig durch einen Operator eliminiert wird bei relativ geringen Verzerrungen des Objekts. Hier wurde eine Maskenoperation angewendet, bei der nicht der fünfte, sondern der dritte Histogrammwert dem Zentralpunkt als neuer Wert zugewiesen wird. Beide genannte Operatoren gehören zur Klasse der Rangordnungsoperatoren $R_{N,n}$; N bezeichnet die Größe der quadratischen $N \times N$-Maske und n den Rang in der geordneten Folge von $N \times N$ Grauwerten, dessen Wert als neuer Grauwert des Zentralpunktes benutzt wird. Wird ein niedriger Rang benutzt, werden größere Störflecken unterdrückt; gleichzeitig tritt jedoch eine verstärkte Erosion der Kontur auf. Rangordnungsoperatoren sind damit geeignet, periodische oder linienförmige Strukturen im Bild zu unterdrücken, wie sie z.B. durch Verwendung von Transportnetzen bei der Durchlichtmethode auftreten sowie Kratzer, Rillen und Verschmutzungen zu beseitigen, wenn der Rang entsprechend gewählt wird.

Weitere Maskenoperatoren werden zur Bildung sauberer Schwarzweißübergänge aus dem Grauwertbild (Kantenverschärfung, Kantendetektion), zur Bestimmung eines eindeutigen Binärübergangs und zur Umfangsbestimmung aus dem Binärbild eingesetzt. Einige dieser Operatoren und ihre Wirkung sind in Tafel 7.2 dargestellt.

Maskenoperationen werden aufgrund ihrer zyklischen Arbeitsweise zur Erzielung hoher Verarbeitungsgeschwindigkeiten zweckmäßigerweise hardwaremäßig implementiert.

Für die Merkmalsgewinnung bei der 2-D-Erkennung wird ein binäres Bild benötigt. Zur Binarisierung des von der Kamera erzeugten und evtl. schon verbesserten grauwertigen Bildes ist eine Schwelle S erforderlich, die jeden Bildpunkt entweder der Objektbildpunktmenge oder der Untergrundbildpunktmenge zuordnet. Eine intuitive Festlegung der Schwelle S ist nur für sehr großen Objekt-Untergrund-Kontrast möglich. Zur Selek-

Tafel 7.2. Beispiele für Maskenoperationen mit 3 × 3-Fenster

Operator		Wirkungsweise	Anwendung
Median-operator	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	Auswahl eines Grauwertes	Bildsäuberung und -glättung
Laplace-Operator	$\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}$	bewertete Summe	Kanten-detektion
Deblurring-Operator	$\begin{bmatrix} & -1 & \\ -1 & 5 & -1 \\ & -1 & \end{bmatrix}$	bewertete Summe	Kantenver-schärfung
Robinson-Operator	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ $8 \times 45°$	Auswahl eines Maskenwertes	Kanten-detektion
Gradin-Operator	$\begin{bmatrix} & -1 & \\ & 1 & \\ & & \end{bmatrix}$ $8 \times 45°$	Auswahl eines Maskenwertes	Kanten-detektion
Sobel-Operator	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ $2 \times 90°$	Auswahl eines Maskenwertes	Kanten-detektion
Kontur-operator	$\begin{bmatrix} & & \\ 1 & 0 & \\ 1 & & \end{bmatrix}$ $4 \times 90°$	Zählen binärer Maskenwerte	Bestimmung des Konturumfangs
Verfolgungs-operator	$\begin{bmatrix} & & \times \\ & \times & \times \\ & & 0 \end{bmatrix}$ $8 \times 45°$	Auswahl der Masken-verschiebung	Kontur-verfolgung

tion optimaler Schwellen wurden zahlreiche Methoden entwickelt; hier soll nur die Histogrammethode dargestellt werden. Diese beruht auf der Auswertung der Häufigkeitsverteilung der Grauwerte des gesamten Bildes. Ordnet man jedem Grauwert die Anzahl der Bildpunkte zu, die diesen Grauwert enthalten, und ordnet diese in der Reihenfolge der Grauwerte, so entsteht ein Histogramm, das bei gutem Objekt-Hintergrund-Kontrast zwei Maxima enthält. Legt man die Schwelle in den Grauwert, der dem zwischen den Maxima liegenden Minimum entspricht, ist eine gute Trennung zwischen Objektbildpunkten und Hintergrundbildpunkten möglich (Bild 7.35a).

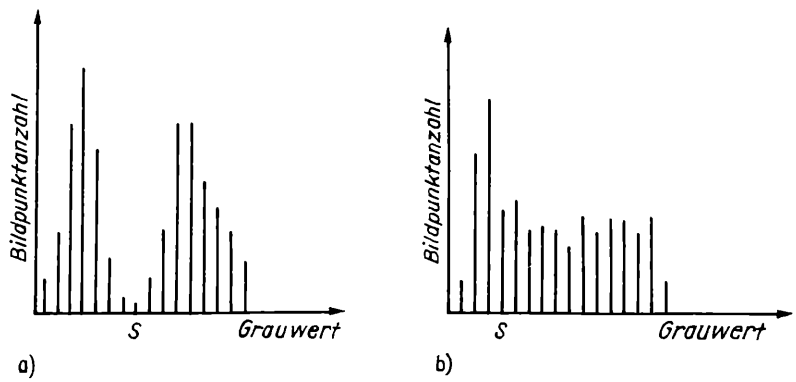


Bild 7.35. Bildhistogramme mit 16 Graustufen und Festlegung der Binarisierungsschwelle
a) nach der Minimummethode; b) anhand einer Grauwertschulter

Häufig läßt sich dieses Minimum jedoch nur schwer lokalisieren. Bild 7.35b zeigt die Selektion einer Schwelle mittels einer sog. Grauwertschulter.

Treten im Histogramm mehrfach Minima oder Grauwertschultern auf, lassen sich mehrere Schwellen selektieren. Die Zugehörigkeit des betreffenden Bildpunktes zum Objekt oder Untergrund wird danach anhand der Punktumgebung durch entsprechende Maskenfilterung oder andere Klassifikationsalgorithmen entschieden [7.22].

Nach der Erstellung des Binärbilds, die in der Regel durch eine angepaßte Hardware erfolgt, ist dieses zur weiteren Verarbeitung im Mikrorechner zwischenzuspeichern. Die vollständige Abspeicherung erfordert $n_x n_y$ Speicherplätze. Die Anzahl der erforderlichen Speicherplätze kann für wenig strukturierte Bildinformation wesentlich reduziert werden, wenn anstelle aller Bildpunkte nur diejenigen abgespeichert werden, die eine echte Information enthalten. In unserem Fall sind dies die Schwarzweißübergänge. Es ist also ausreichend, nur diese Übergänge als Adresse in x- und y-Richtung anzugeben. Ist ein Objekt vollständig im Bildformat enthalten, kann beim Abtasten einer Bildzeile auf einen Schwarzweißübergang nur ein Weißschwarzübergang folgen und umgekehrt. Ein Verfahren, das diese Tatsache zur Bildspeicherung nutzt, ist das sog. „run-length-coding". Hier wird lediglich die erste Adresse eines Übergangs und die folgende Bildpunktzahl mit gleichem Binärwert gespeichert.

7.4.3. Merkmalsgewinnung und Klassifikation

Aus dem 2dimensionalen Binärbild lassen sich morphometrische Merkmale berechnen, die die Abbildung in verkürzter Form beschreiben – sie werden auch als „Repräsentanten" des Objekts bezeichnet. Die zu erkennenden Teile besitzen i. allg. je nach momentaner Lage unterschiedliche Silhouettenbilder. Diese werden bei der Auswertung wie

unterschiedliche Objekte behandelt, d.h., zu jedem möglichen Silhouettenbild sind die zugehörigen Repräsentanten in der Lernphase zu bestimmen und in der Erkennungsphase anhand der abgespeicherten Modelldaten zu vergleichen. Die Erkennung ist damit nur möglich, wenn das Teil eine endliche Zahl von stabilen Lagen auf der Unterlage einnimmt oder die Merkmale gegenüber silhouettenverändernden Drehungen invariant sind (Rotationskörper).

Unter der Voraussetzung, daß sich die Repräsentanten aller Objekte des betrachteten Teilesortiments – also aller Teile in allen stabilen Lagen – voneinander unterscheiden, liefert der Erkennungsprozeß zwei Aussagen: zum einen die Teileklasse und zum andern die Lageklasse. Die Art der auszuwählenden Merkmale richtet sich damit nach der Erkennungsaufgabe – es sind diejenigen Merkmale zu benutzen, die die Separierung der Objekte gewährleisten.

Einige für die Teileerkennung häufig benutzte, direkt berechenbare oder abgeleitete Merkmale sind:

(D_x, D_y) Abmessungen des achsenparallelen Rechtecks, das das Objekt in der momentanen Lage umschreibt,
abgeleitet: D_g Diagonale dieses Rechtecks

F Flächeninhalt des Objekts mit oder ohne Lochflächen,
abgeleitet: $\sqrt{F}$, D_g/F, D_g^2/F

Kp Anzahl der Konturpunkte des Objekts (einschließlich bzw. ausschließlich der Konturpunkte von Löchern im Objekt)

U Umfang des Objekts (einschließlich bzw. ausschließlich des Umfangs von Löchern),
abgeleitet: Fok $= 4\pi F/U^2$ Kreisformfaktor
$\sqrt{F}/U$ Kompaktheit

Z Anzahl der Löcher im Objekt

$L_f(i)$ Fläche des i-ten Loches,
abgeleitet: Summe aller Lochflächen
Verhältnis der Summe aller Lochflächen zur Gesamtlochzahl usw.

Weitere wichtige Merkmale lassen sich aus den Flächenträgheitsmomenten gewinnen. Sind s_x und s_y die Rasterkoordinaten des Flächenschwerpunktes, so gilt für die Zentralmomente bezüglich der x- und y-Achse

$$T = \sum_{y=1}^{m} \sum_{x=1}^{n} (x - s_x)^a (y - s_y)^b B(x, y). \tag{7.6}$$

$B(x, y)$ ist eine Belegungsfunktion; für einen Objektpunkt ist $B = 1$, sonst Null. Innerhalb der Lochfläche kann $B = 1$ bzw. Null gesetzt werden, je nachdem, ob die Lochfläche berücksichtigt werden soll oder nicht. Mit $a = b = 0$ wird die Objektfläche gewonnen. Für $a = 1, b = 0$ und $a = 0, b = 1$ ergeben sich die Schwerpunktkoordinaten. Außerdem ergeben sich mit

$a = 2, b = 0$ T_{xx} quadratisches Flächenträgheitsmoment bezüglich der x-Achse

$a = 0, b = 2$ T_{yy} quadratisches Flächenträgheitsmoment bezüglich der y-Achse

sowie das polare Flächenträgheitsmoment

$$T_p = T_{xx} + T_{yy}. \tag{7.7}$$

Diese Flächenträgheitsmomente sind orientierungsabhängig, ebenso alle weiteren Momente höherer Ordnung ($a + b > 2$). Aus den Flächenträgheitsmomenten bezüglich der Rasterkoordinatenachsen lassen sich die orientierungsunabhängigen Hauptträgheitsmomente des Objekts (Objekt bedeutet in diesem Zusammenhang immer Binärbild, also Fläche) bestimmen:

$$T_{tt}, T_{ss} = T_p/2 \pm \sqrt{(T_{xx} - T_{yy})^2/4 + T_{xy}^2}. \tag{7.8}$$

T_{xy} ergibt sich aus Gl. (7.6) durch $a = 1$, $b = 1$.

Wurde der Flächenschwerpunkt (s_x, s_y) bestimmt, lassen sich einige von diesem ausgehende ausgezeichnete Radien zu den Konturpunkten als Merkmale bestimmen. Sind $r_x(i)$, $r_y(i)$ die Rasterkoordinaten der Konturpunkte, gilt allgemein für die vom Flächenschwerpunkt ausgehenden Objektradien

$$R(i) = \sqrt{(r_x(i) - s_x)^2 + (r_y(i) - s_y)^2}. \tag{7.9}$$

Durch Vergleich lassen sich die extremalen Radien R_{min} und R_{max} sowie der mittlere Objektradius bestimmen.

Verfolgt man von einem Punkt ausgehend die Kontur und ermittelt den zugehörigen Objektradius, so tritt bei bestimmten konkaven Konturverläufen eine sog. Rückläufigkeit des Radiuswinkels auf, die ebenfalls als Merkmal dienen kann [7.23] (Bild 7.36).

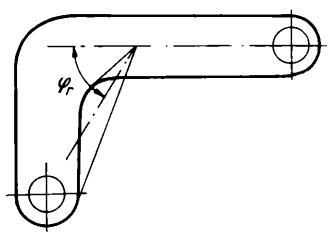


Bild 7.36
Rückläufigkeit des Radiuswinkels als Erkennungs- und Orientierungsmerkmal

Ausschlaggebend für die Auswahl von Merkmalen ist neben den Eigenschaften der zu erkennenden Objekte die Laufzeit der zugehörigen Algorithmen auf der verwendeten Rechentechnik. Auf die zahlreichen Methoden zur effektiven Merkmalsberechnung und die verwendeten Algorithmen, die zum großen Teil auf Maskenoperationen sowie rechentechnisch einfach realisierbaren Operationen, wie Verschiebung, Vergleich, Addition und Subtraktion, beruhen, soll hier nicht näher eingegangen werden. Eine umfangreiche Literaturzusammenstellung findet sich in [7.24] [7.26].

Der Klassifikator vergleicht die aus dem Objekt ermittelten Merkmale mit denen der abgespeicherten Modellobjekte und ordnet das Objekt in die zugehörige Objektklasse ein. Sind unterschiedliche Teilelagen zugelassen, entspricht die Objektklasse jeweils einer stabilen Lage des zu erkennenden Teiles; bei k stabilen Lagen sind für ein zu erkennendes Teil k Objektklassen, für n Teile nk_t Objektklassen zu definieren. Sollen die Teile in beliebiger Position und Orientierung erkannt werden, können nur positions- bzw. orientierungsunabhängige Merkmale zur Klassifikation benutzt werden. An den Klassifikator sind gewisse Robustheitsforderungen zu stellen, die die möglichen Teiletoleranzen, Abbildungsfehler und bei der Vorverarbeitung und Merkmalsbildung auftretende Rundungsfehler betreffen. Der Aufbau von Klassifikatoren kann dabei nach zwei unterschiedlichen Prinzipien erfolgen.

Bei der *parallelen* Klassifizierung werden alle Merkmale gleichzeitig zum Vergleich angeboten. Hierzu ist es erforderlich, daß alle zur Unterscheidung der Objekte benötigten Merkmale ausreichend schnell bestimmt werden können. Sind statistisch erfaßbare Stö-

rungen bei der Ermittlung der Objektmerkmale von Bedeutung, ist ein statistisches Klassifikationsverfahren anzuwenden. Einen optimalen Klassifikator liefert dann die Bayes-Regel, ergänzt durch Rückweisungsbedingungen, wenn das Objekt in keine der vorgesehenen Klassen einordenbar ist [7.25] [7.26].

Die *sequentielle* Klassifikation beruht auf der Tatsache, daß zur Erkennung der Objekte nicht immer alle Objektmerkmale herangezogen werden müssen. Es werden durch eine Interaktion zwischen Klassifikator und Merkmalsbestimmung nur diejenigen Merkmale berechnet, die zur Einordnung der Objekte notwendig sind. Hierdurch ist in der Regel eine erhebliche Reduktion des Rechenaufwands möglich.

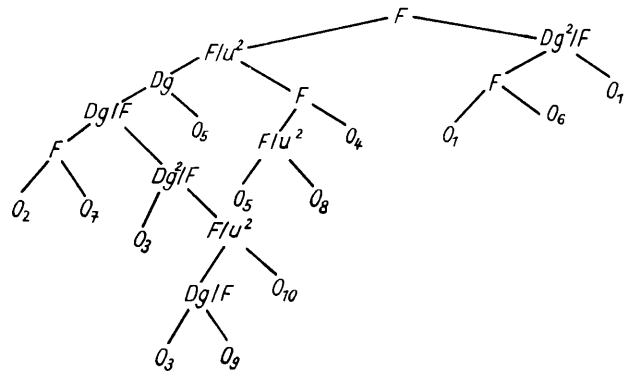


Bild 7.37
Beispiel eines Entscheidungsbaums zur Objektklassifikation

Eine Variante zur sequentiellen Klassifikation ist der Aufbau sog. Entscheidungsbäume, die nachfolgend an einem einfachen Beispiel erläutert werden soll. Die Knoten des im Bild 7.37 dargestellten Entscheidungsbaums stellen Trennebenen für die Merkmale dar. Ist das berechnete Objektmerkmal größer als ein in der Lernphase eingegebener Wert, wird der rechte nachfolgende Knoten berechnet; ist es kleiner, der linke. Eine Berechnung der absoluten Merkmalsabweichung entfällt, d.h., der Algorithmus wird sehr schnell arbeiten. Die Knoten werden nacheinander abgearbeitet, bis der Endknoten erreicht ist; er entspricht der ermittelten Objektklasse. Solche Entscheidungsbäume können sehr effektiv der geforderten Erkennungsgenauigkeit und der Anzahl der zu unterscheidenden Teile angepaßt werden. In diesem Beispiel wurde auch ein Merkmal benutzt, das nicht vollständig orientierungsunabhängig ist, nämlich die Diagonale des umschreibenden Rechtecks. Diese läßt sich aber sehr effektiv berechnen und dient häufig der Unterscheidung von Teilen mit unterschiedlicher Kompaktheit. Um ein Teil, das auf der Ablagefläche eine beliebige Position und Orientierung einnehmen kann, mit dem Manipulator richtig zu greifen, ist das Frame T_g in eine bestimmte Relation zum objekteigenen Koordinatensystem zu bringen, in dem die Greifpunkte angegeben sind. Zur Bestimmung des Positionsvektors $\boldsymbol{p}$ kann der Flächenschwerpunkt (s_x, s_y) herangezogen werden. Wesentlich aufwendiger ist die Berechnung des Orientierungswinkels Φ. Prinzipiell ist durch Ermittlung des Winkels, den ausgezeichnete Objektradien mit den Achsen des Rasterfeldes bilden, eine Orientierungsbestimmung möglich. Von großer Genauigkeit und Robustheit ist beispielsweise die Berechnung des Winkels zwischen der Achse des kleinen Hauptträgermoments T_{ss} und der x-Achse des Rasterfelds, der aber nur mod 180° angebbar ist [7.27] [7.28]. Die erreichbare Genauigkeit ist neben der Streuung der Trägheitsmomente auch vom Unterschied der beiden Hauptträgheitsmomente abhängig. Um die Mehrdeutigkeit zu eliminieren, können zusätzliche Winkel mit relativ geringer Genauigkeit bestimmt werden, z.B. der Winkel, den der Vektor zwischen Flächenschwer-

punkt und Konturschwerpunkt mit der x-Achse bildet, der Winkel des Vektors zwischen Flächenschwerpunkt und Lochschwerpunkt sowie der x-Achse usw. Eine weitere Methode beruht auf der Teilebeschreibung mit Radialschnitten. Vom zu erkennenden Teil wird gemäß Bild 7.38 ein Radialprofil in justierter Orientierung aufgenommen. Das Radialprofil besteht aus Strecken vom Flächenschwerpunkt (s_x, s_y) zu den Schnittstellen der Radien mit der Kontur. Die Radien sind auf einer vorgegebenen, meist gleichmäßigen Winkelteilung angeordnet. Zur Ermittlung des Orientierungswinkels wird in der Erkennungsphase das gemessene Radialprofil so lange gegenüber dem Modellprofil verschoben, bis maximale Übereinstimmung herrscht.

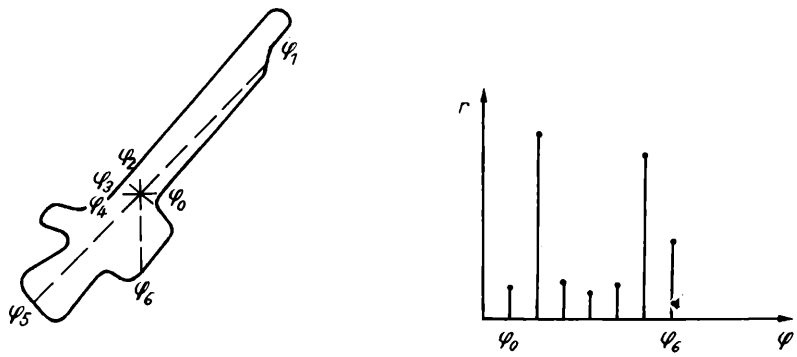


Bild 7.38. Orientierungsbestimmung mit Radialprofil

Die erreichbare Genauigkeit hängt dann von der Winkelteilung des Profils ab. Das Radialprofil wird ebenfalls für die Erkennung der Objekte eingesetzt.

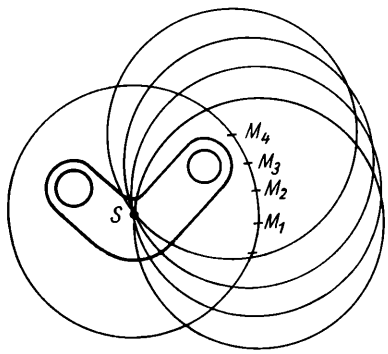


Bild 7.39
Methode der polaren Kodierung

Ein aus dem Radialprofil abgeleitetes, algorithmisch schnell zu realisierendes Verfahren benutzt das Prinzip der polaren Kodierung [7.29]. Um den Flächenschwerpunkt wird gemäß Bild 7.39 ein Kreis K_0 mit dem Radius R gelegt. Das Objekt wird dann mit einer Kreisschar mit Mittelpunkten auf K_0 und gleichem Radius R abgetastet; diese Kreise schneiden alle den Flächenschwerpunkt. Für jeden Kreis K_i wird die Anzahl der Objektbildpunkte aufgenommen. Das erhaltene Profil wird mit dem Referenzprofil verglichen und die Orientierung bestimmt.

Sind die zu erkennenden Teile so beschaffen, daß sie durch die Wirkung eines mechanischen Anschlags nur eine geringe Zahl stabiler Vorzugsorientierungen annehmen, kann die Orientierungsbestimmung häufig auf eine Lageerkennung zurückgeführt werden. Das Prinzip kann insbesondere bei Flachteilen vorteilhaft angewendet werden [7.30].

Entsprechend Bild 7.40 werden die Teile auf einem Transportband oder einer Translationseinheit an einem Anschlag ausgerichtet; nach Freigabe der Bewegung (Heben des Anschlags) wird durch die Erkennungseinheit der Flächenschwerpunkt ermittelt und durch einfache Vergleichsverfahren mit den Referenzdaten der möglichen Vorzugsorientierungen sowohl eine vereinfachte Erkennung als auch die Bestimmung der eingenommenen Orientierung durchgeführt. Die Referenzdaten müssen für jede Vorzugsorientierung in der Lernphase eingegeben werden.

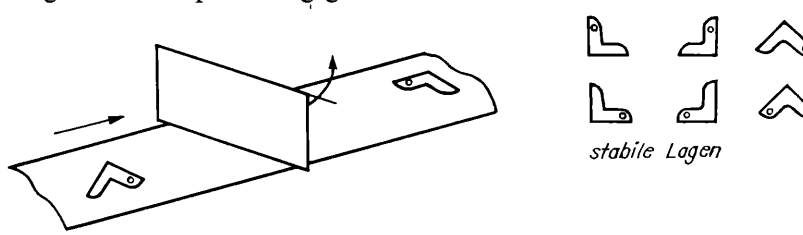


Bild 7.40. Ausrichtung von Flachteilen an einen Anschlag

Für Erkennungsaufgaben mit festen Orientierungen der Objekte läßt sich auch ein einfaches Verfahren zum vollständigen Mustervergleich des Objekts (matching) mit dem Modellobjekt aufbauen, da nur Translationen zur Herstellung der optimalen Überdekkung aller Zeileninhalte notwendig sind.

Mustervergleichsverfahren dienen neben der Erkennung insbesondere der Qualitätsüberwachung, da mit ihnen jede Objektabweichung registriert und bewertet werden kann. Hierzu ist die vollständige Abspeicherung eines Repräsentanten der jeweiligen Objektklasse erforderlich. Beim Mustervergleich der Silhouetten, auch Schablonenvergleich (template matching) genannt, wird praktisch jeder Bildpunkt als Objektmerkmal ausgewertet.

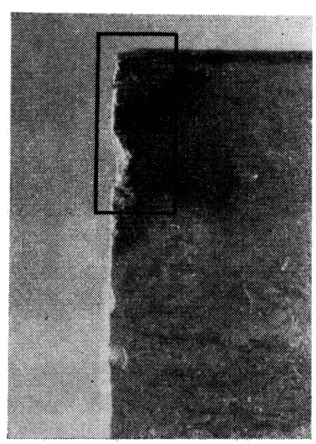
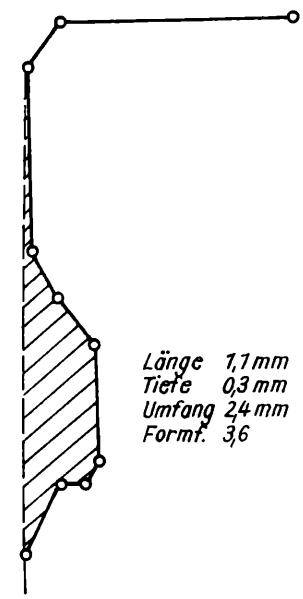


Bild 7.41
Fotografie einer durch Seitenlicht aufgehellten Kontur

Die optische Erkennung und Bewertung einzelner Werkstückstrukturen (Kanten, Schweißfugen, Grat, Werkstückfehler) erfordert i. allg. aus technologischen Gründen die Anwendung der Auf- oder Schräglichtmethode und damit eine Grauwertverarbeitung des Rasterbilds. Das Problem besteht darin, aus häufig recht komplizierten Grauwertübergängen an den Objektstrukturen auswertbare Schwarzweißübergänge zu erzeugen, die mit der gesuchten Struktur am besten übereinstimmen. Die günstige Anordnung der Beleuchtung unter den konkreten technologischen Bedingungen ist hierfür eine wichtige Voraussetzung.

Die Folge einer Bildverarbeitung vom Grauwertbild bis zur Darstellung und Bewertung der Kantenstruktur nach *Roesler* [7.31] wird im Bild 7.42 gezeigt.

a)	b)	c)
.....	.....	.....
.....15BFFFFFFFF	.....15BFFFFFFFF	65555555
.....5AEFFFFFFFF	.....5994.....	.....65.....
.....3AEFFFFFFFF	.....3741.....	.....7.....
.....3AEFFFFFFFF	.....3741.....	.....7.....
.....17CFFFFFFFFF	.....17951.....	.....7.....
.....17CFFFFFFFFF	.....16641.....	.....7.....
.....16BEFFFFFFFF	.....16631.....	.....7.....
.....5BEFFFFFFFF	.....5731.....	.....7.....
.....4BFFFFFFFFF	.....4741.....	.....7.....
.....4BFFFFFFFFF	.....4861.....	.....7.....
.....39EFFFFFFFF	.....3961.....	.....8.....
.....8EFFFFFFFF	.....8B61.....	.....8.....
.....39EFFFFFFF	.....37B93.....	.....8.....
.....236CFFFFFF	.....2249B61...	.....18.....
.....12349EFFFF	.....12116A63..	.....7.....
.....23349CFFF	.....21.1533..	.....7.....
.....23469CFFF	.....2113533..	.....7.....
.....13569CFFFF	.....13335663..	.....7.....
.....1358ACFFFF	.....1133433...	.....7.....
.....1368BEFFFF	.....1243343...	.....6.....
.....14668CFFFFF	.....14532441...	.....655.....
.....159DFFFFFFFF	.....15899973....	.....6.....
.....59DFFFFFFFFF	.....58862.....	.....6.....
.....159EFFFFFFFFF	.....15452.....	.....7.....
.....159EFFFFFFFFF	.....14451.....	.....7.....



d)

Bild 7.42. Folge einer Bildverarbeitung bis zur Bewertung des im Bild 7.41 enthaltenen Objekts durch morphometrische Merkmale nach Roesler [7.31]

a) Ausschnitt des digitalisierten Grauwertbildes, dessen 16 Graustufen durch Hexadezimalzahlen dargestellt sind
b) Detektion der optischen Kante aus dem Grauwertbild durch einen Gradin-Operator (Tafel 7.2) – die Kante wird durch ein Grauwertgebirge dargestellt
c) Spur der optischen Kante, die durch einen Verfolgungsoperator erzeugt wurde – die Spur wird durch eine Folge von Freemanschen Richtungszahlen, einer Freeman-Kette, kodiert, und die Freeman-Zahlen 1 bis 8 bezeichnen die acht möglichen Nachbarpunkten gehörenden Richtungen
d) die Freeman-Kette beschreibt eine geordnete und dichte Punktfolge, die nach einer Geradenapproximation durch einen Polygonzug dargestellt wird – Abweichungen von einer Sollgeraden können durch Merkmale bewertet werden

Aus dem digitalisierten Grauwertbild (b) wird durch Anwendung des Gradin-Operators (Tafel 7.2) ein Grauwertgebirge in der Übergangszone erzeugt (c). Mit Hilfe des Konturfolgeoperators wird hieraus eine Spur ausgewählt, die den geringsten Abweichungen in den Bildzeilen folgt; diese wird als Kette von Freemanschen Richtungszahlen kodiert. Gleiche Zahlen in der Kette bedeuten, daß in der darunterliegenden Zeile die Spur ihre Richtung beibehält; Richtungsabweichungen werden durch Zahlen von 1 bis 8 dargestellt.

Mit den Freemanschen Richtungszahlen ist dann eine Geradenapproximation möglich, die die Kurvenbeschreibung auf je einen Punkt für ein Geradenstück (beim ersten Geradenstück zwei Punkte) reduziert (d). Wird die Erkennung zur Qualitätssicherung benutzt, kann die Strukturabweichung durch Vergleich mit einer Referenzstruktur (im Bild als Gerade dargestellt) durch weitere Merkmale bewertet werden.

Abschließend wird anhand vorgegebener Qualitätsmerkmale geprüft, ob diese im Toleranzbereich liegen.

Die ermittelten Konturdaten können auch direkt zur Manipulatorsteuerung benutzt werden. In [7.32] ist eine Pilotlösung beschrieben, bei der die Schweißnahtverfolgung und Steuerung des Schweißprozesses direkt nach visuell ermittelten Konturdaten erfolgen.

7.4.4. Visuelles System zur Teileerkennung und Lagebestimmung

Bei der Realisierung visueller Erkennungssysteme wird in der Regel ein Kompromiß zwischen Funktionsumfang, Bedienungskomfort, Universalität einerseits sowie dem hierzu erforderlichen Aufwand und damit Preis andererseits zu schließen sein. Wichtige Kriterien für den praktischen Einsatz visueller Erkennungssysteme sind der Umfang des zu erkennenden Teilesortiments, die Größe der Abweichungen der Teile untereinander, der Wechsel von Teilesortimenten (Änderung der Aufgabe) sowie die vorliegenden technologischen Möglichkeiten. Ist beispielsweise im technologischen Prozeß ein Teiletransport sowieso notwendig, läßt sich eine Zeilenkamera i. allg. mit geringerem Aufwand installieren als eine Matrixkamera. Zu beachten sind außerdem die Bedingungen für eine günstige Beleuchtungstechnik sowie aus der vorhergehenden Bearbeitungsstufe resultierende Störeinflüsse (Verschmutzungen, Späne, Kühlmittelfilm).

Visuelle Erkennungssysteme mit Zeilen- und Matrixkamera unterschiedlichen Auflösungsvermögens und Funktionsumfangs werden seit einigen Jahren industriell produziert und praktisch eingesetzt [7.34] [7.35] [7.36]. Für die Vorverarbeitung, Merkmalsbildung und Klassifikation sind neben Spezialhardware Klein- und Mikrorechner verschiedener Konfigurationen vorgesehen. Derzeit mit hoher Erkennungssicherheit beherrschte Aufgaben sind folgende:

Sortieren und Beschicken

Unterschiedliche Teile werden in der Regel vereinzelt auf einem Transportband unter einer Kamera vorbeigeführt und nach der Erkennung der Teileklasse, der Position und Orientierung von einem IR in der ermittelten teilespezifischen Greifposition gegriffen und in der für das Teil erforderlichen Position abgelegt. Das Greifen erfolgt dabei sowohl bei gestopptem als auch bei bewegtem Transportsystem.

Ermittlung von Konturverläufen und Werkstückfehlern

Die erreichbare Genauigkeit der bereits beschriebenen Verfahren hängt dabei wesentlich von der beleuchtungsbedingten Art der Silhouettenbildung und dem Auflösungsvermögen der Kamera ab.

Bestimmung von Relativpositionen

Ein wichtiges Anwendungsgebiet ist hier die Montage; durch die Erkennung von geforderten Montagepositionen kann der Aufwand in der Peripherie verringert werden. Beispiele sind das Beschicken elektronischer Leiterplatten und Einlegearbeiten.

Ein weiteres wichtiges Einsatzgebiet zukünftiger Erkennungssysteme ist die visuelle Inspektion von Werkzeugen für bedienerlose oder bedienarme Fertigungsprozesse. Durch Erkennung von Werkzeugbruch und -abnutzungsgrad muß hier Nachfolgeschaden vermieden werden. Die Vielfalt auftretender Verschleißerscheinungen und ihre Bewertung bedingen komplizierte Verarbeitungsverfahren und umfangreiches Modellwissen zur objektiven Entscheidung über die weitere Verwendbarkeit von Werkzeugen, für die auch der Bediener oft langjährige Erfahrung benötigt.

Nachfolgend wird eine Pilotlösung vorgestellt, in der ein visuelles Erkennungssystem auf der Basis einer Zeilenkamera und eines 8-bit-Mikrorechners mit einem IR ZIM 10 zum Sortieren von Werkstücken gekoppelt wurde. Die Hardwarekonfiguration wird im

Bild 7.43 gezeigt. Als Bildwandler dient die CCD-Zeile L 110 [7.17]; weitere Bestandteile der Kamera sind die Takterzeugung, die Steuerung der Binarisierungsschwelle und der Integrationszeit im Bildwandler zur Einstellung der optimalen Empfindlichkeit und des Auflösungsbereichs der Amplituden (s. Abschn. 7.4.1.) sowie eine Hardwarelösung zur Erzeugung des Run-length-Kodes [7.37].

Die Teilebeleuchtung erfolgt mittels Streifenschräglichts; als Beleuchtungsquellen dienen Halogenstablampen in Verbindung mit Kunststoffzylinderlinsen. Mit dieser Beleuchtungstechnik wird ein guter Kontrast auch bei geringen Reflexionsfaktordifferenzen zwischen Werkstück und Auflage erreicht, so daß eine Binärbildverarbeitung vorgenommen werden kann. Die Werkstücke werden auf einer linearen Verfahreinheit unter der Kamera vorbeibewegt. Die Binärbilddaten gelangen über einen Multiplexer in den Mikrorechner und werden zwischengespeichert. In einem ersten Verarbeitungsschritt wird durch ein Bildrahmenprogramm getestet, ob das Binärbild vollständig im x-y-Raster enthalten ist, d.h., es dürfen keine Schwarzweißübergänge an den äußeren Punkten des Rasters auftreten. Weitere RAM-Speicher sind zur Abspeicherung der Repräsentanten der Teileklassen (Schablone) und für die Erkennungs- und Schablonenvergleichsalgorithmen vorgesehen.

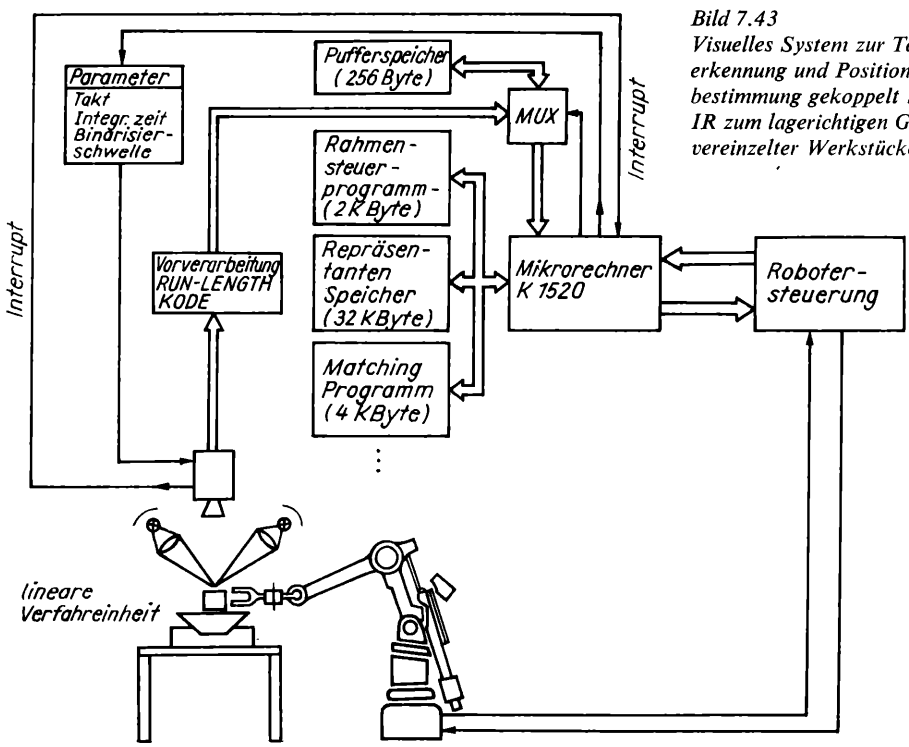


Bild 7.43
Visuelles System zur Teileerkennung und Positionsbestimmung gekoppelt mit einem IR zum lagerichtigen Greifen vereinzelter Werkstücke

Die Programmstruktur wird im Bild 7.44 gezeigt. Für den Erkennungsprozeß sind drei Phasen charakteristisch:

In der *Trainingsphase* erfolgt die Belehrung des Systems durch Vorzeigen je eines Vertreters des gesamten Teilesortiments; es werden die Merkmale bestimmt und für den Schablonenvergleich ein Repräsentant abgespeichert.

In der *Prüfphase* können unter Führung des Bedieners Programme zur Merkmalsbildung und Klassifikation ausgewählt und modifiziert werden.

In der *Arbeitsphase* erfolgt die Datenübergabe an die Robotersteuerung.

Das Erkennungssystem enthält Module zum Mustervergleich, zur Teileklassifizierung und zur Orientierungsbestimmung, die einheitlich in die Blöcke Trainings-, Prüf- und Arbeitsphase gegliedert sind. Für die Trainings- und Prüfphase ist der Anschluß eines Bildschirmgeräts vorgesehen. Die Programme sind in ein Betriebssystem eingebettet, das eine Führung des Bedieners gestattet.

Der vollständige Mustervergleich ist für Teile geeignet, die durch Ausrichtung an einem Anschlag nur einige Orientierungen einnehmen können. Die Schwarzweißübergangsbilder dieser Orientierungen werden von allen Teilen in der Trainingsphase gespeichert. In der Arbeitsphase werden durch Schwerpunktzentrierung von Bild und Repräsentant das Teil erkannt, seine Orientierung und die Flächenschwerpunktkoordinaten bestimmt.

Die Teileklassifizierung arbeitet mit morphometrischen Merkmalen; für die Klassifikation wird ein Entscheidungsbaum verwendet. Die Orientierungsbestimmung erfolgt nach der Teileklassifikation durch Richtungsmerkmale (Trägheitsmomentachsen, Vektoren zu Konturpunkten usw.). Durch den Bediener wird aus diesen Modulen eine der zu lösenden Erkennungsaufgabe adäquate Auswahl getroffen.

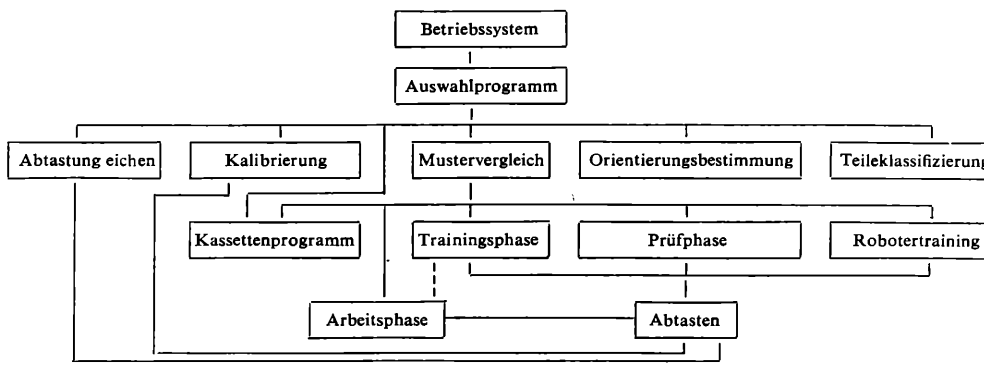


Bild 7.44. Modulares Programmsystem der Erkennungseinheit nach Bild 7.43

Weitere Programmteile dienen der Eichung der Kamera, der Abtastung sowie der Einstellung eines quadratischen Rastermaßes (gleiche Bildpunktabstände in x- und y-Richtung bei bewegtem Teil).

Der Modul Kalibrierung realisiert die Transformation der Bildkoordinaten in das kartesische Basiskoordinatensystem des Roboters. Da die z-Koordinaten unverändert bleiben, sind nur die variablen x- und y-Koordinaten sowie ein Orientierungswinkel für das jeweils erkannte Teil an die Robotersteuerung zu übergeben. Die Trainingsphase beinhaltet auch das Teachen der Greiferstellung des IR. Von der Robotersteuerung werden die (x, y)-Koordinaten des Greifpunktes und der Orientierungswinkel im Basiskoordinatensystem übergeben. Mit vom Erkennungssystem ermitteltem Flächenschwerpunkt und Orientierungswinkel des Teiles werden der Abstand des Greifpunktes vom Schwerpunkt und der Winkel zwischen der Greiferorientierung und Teileorientierung ermittelt und im Erkennungssystem gespeichert. In der Arbeitsphase werden an die IR-Steuerung die Teileklasse, die aktuellen (x, y)-Koordinaten des Greifpunktes und der aktuelle Winkel Φ in Basiskoordinaten übergeben; die Verschiebung durch das Transportsystem wird dabei verrechnet. Die Teileklasse dient dem Aufruf des für das Teil spezifischen Unterprogramms zur Ablage auf einer Palettenposition. Der Datenaustausch zwischen Erkennungssystem

und Robotersteuerung erfolgt über eine serielle Standardschnittstelle. Die erreichten Erkennungszeiten liegen unter einer Sekunde.

Das beschriebene Erkennungssystem wurde im Zentralinstitut für Kybernetik und Informationsprozesse der AdW entwickelt; die Pilotlösung wurde durch ein gemeinsames Forschungskollektiv der AdW, des VEB Zentraler Ingenieurbetrieb der Metallurgie (ZIM) und des Werkzeugmaschinenkombinats „7. Oktober" realisiert.

7.4.5. 3-D-Erkennung

Die fehlende Tiefeninformation wird bei der 2-D-Erkennung durch Einschränkungen der zulässigen Parametervariation und des erkennbaren Teilesortimentes ersetzt.

Tiefeninformation kann auch mit 2-D-Erkennungssystemen durch Verwendung strukturierten Lichtes oder Ausnutzung der Perspektive gewonnen werden.

Wird durch einen Projektor ein Parallelstrahl erzeugt, der nicht in der optischen Achse des Bildaufnahmesystems liegt, so kann aus dem Abbildungsort des von einem Objekt reflektierten Bildpunktes auf dessen Entfernung geschlossen werden (Triangulation).

Die Ausnutzung der Perspektive ist möglich, wenn eine größenunabhängige Teileerkennung durchgeführt wird und durch Vergleich von Abständen markanter Punkte des Objekts mit denen des gespeicherten Repräsentanten die Veränderung des Abbildungsmaßstabs bestimmt wird. Beide Verfahren rechnen zu den sog. $2\frac{1}{2}$-D-Systemen [7.39].

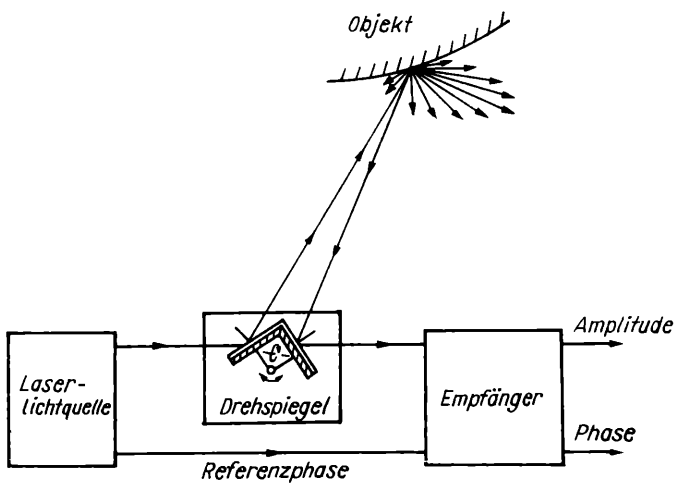


Bild 7.45
3-D-Abtastung eines Objekts mittels Laserstrahls

Eine echte 3-D-Erkennung erfordert entweder den Einsatz von zwei Kameras und Triangulation der Bildpunkte oder die Messung des vom Licht zwischen Reflexionspunkt und Kamera zurückgelegten Weges über eine Laufzeitmessung. Das Prinzip der Laufzeitmessung ist im Bild 7.45 dargestellt. Eine Laserlichtquelle sendet amplitudenmoduliertes Licht auf einen in zwei orthogonalen Richtungen drehbaren Spiegel mit senkrecht zueinander angeordneten Reflexionsflächen. Im Empfänger wird mittels eines Phasendetektors die Laufzeit bestimmt. Die Stellung des Drehspiegels bestimmt die Lage des Bildpunktes im Raster; aus der Laufzeit kann bei Berücksichtigung der Spiegelstellung die Entfernung des Objekts ermittelt werden.

An der Realisierung praktisch einsetzbarer 3-D-Erkennungssysteme wird intensiv gearbeitet. Dies betrifft sowohl die Verbesserung der Bildaufnahmetechniken als auch die Methoden der Auswertung. Die Erkennungszeiten für ein 3-D-Datenmassiv sind derzeit noch so groß, daß ein Echtzeitbetrieb nicht möglich ist.

8. Industrielle Steuereinrichtungen

8.1. Allgemeines

In diesem Kapitel werden Industrierobotersteuerungen beschrieben, die aus dem System 600 und dem System ursalog 5000 hervorgegangen sind. Beide bauen auf ein standardisiertes Mikrorechnersystem auf.

Das System 600 des VEB Numerik „Karl Marx" ist auf die Belange der Werkzeugmaschinenindustrie orientiert; zwei aus diesem System abgeleitete Varianten stellen reine Industrierobotersteuerungen dar.

Aus dem System ursadat 5000 des VEB EAW Berlin ist eine Steuerung entstanden, die insbesondere für im direkten Teach-in programmierbare Roboter geeignet ist.

Moderne industrielle Steuereinrichtungen werden heute fast ausschließlich unter Verwendung von einem oder mehreren Mikrorechnern realisiert. Ausnahmen bilden kleine Ablaufsteuerungen, deren Funktionsumfang durch Verbindungsprogrammierung festgelegt wird [8.26]. Die Anwendung von Mikrorechnern hat den Systemgedanken bei der Entwicklung von industriellen Steuereinrichtungen stark geprägt. Dies gilt sowohl für die Konfigurierung eines Systems aus Hardwarekomponenten als auch für die Erstellung der innen, das Funktionalverhalten bestimmenden Software. In der DDR wird für industrielle Steuereinrichtungen der derzeitigen Generation im wesentlichen das vom VEB Kombinat Robotron gefertigte Mikrorechnersystem K1520 eingesetzt. Dieses umfaßt neben den Rechner- und Speicherbaugruppen eine Reihe von Anschlußeinheiten für datenverarbeitungstypische periphere Geräte. Für Steuereinrichtungen werden weitere, prozeßtypische Anschlußeinheiten benötigt, die die Kommunikation mit dem zu steuernden oder zu regelnden Prozeß durchführen.

Diese Lücke schließt das System ursadat 5000 des VEB EAW Berlin mit seinen Anschlußeinheiten für analoge und binäre Signalverarbeitung. Diese vor allem unter dem Aspekt der Störsicherheit für den Industrieeinsatz entwickelten Baugruppen arbeiten ebenfalls auf den standardisierten K1520-Bus. Für Steuerungen, bei denen spezielle, nicht im Angebot eines Systems befindliche Anschlußeinheiten benötigt werden, sind diese unter Beachtung der Busparameter vom Steuerungshersteller selbst zu entwickeln und zu fertigen.

8.2. Mikrorechnersystem K1520

Das Mikrorechnersystem K1520 basiert auf dem Mikroprozessorschaltkreis U880 (CPU), dem Schaltkreis für parallele Ein- und Ausgaben U855 (PIO), dem seriellen Eingabe- und Ausgabeschaltkreis U856 (SIO) und dem Zähler-Zeitgeber-Baustein U857 (CTC). Diese in n-Kanal-Silicon-Gate-Technologie hergestellte Schaltkreisfamilie kann der dritten Generation von Mikroprozessoren zugeordnet werden. Als Festwertspeicher wird der Schaltkreis U555 (1 KByte EPROM) eingesetzt, während als Operativspeicher

Tafel 8.1. Komponenten des Mikrorechnersystems K1520

Bezeichnung	Kurzbezeichnung	Chiffre	Hauptparameter
Zentrale Recheneinheit	ZRE	K2521	Mit: CPU-Schaltkreis U880 (Taktgenerator) PIO-Schaltkreis U855 (Anfangsinitialisierung nach Spannungszuschaltung) CTC-Schaltkreis U857 1 KByte statisch RAM bis 3 KByte EPROM (U555) Koppelelektronik für Mehrrechnerbetrieb
		K2522	Wie K2521, ohne Taktgenerator/Anfangsinitialisierung
		K2523	Wie K2521, ohne CTC-Schaltkreis
		K2524	Wie K2522, ohne CTC-Schaltkreis
		K2525	Mit: CPU-Schaltkreis U880 (Taktgenerator) CTC-Schaltkreis U857 (Anfangsinitialisierung nach Spannungszuschaltung) 8 KByte EPROM (U555)
Operativspeicher	OPS	K3520	nMOS-RAM 4 KByte statisch RAM
		K3521	CMOS-RAM 4 KByte statisch RAM (batteriegestützt)
		K3525	nMOS-RAM 16 KByte dynamisch RAM
Operativ-/ Festwertspeicher	OFS	K3620	nMOS-RAM 2 KByte statisch RAM, bis 6 KByte EPROM
Festwertspeicher	PFS	K3820	bis 16 KByte EPROM
Anschlußsteuerung für Folienspeicher	AFS	K5121	Anschluß bis zu 4 Floppy-Disk-Laufwerken des Typs MF 3200
Anschlußsteuerung für Lochbandgeräte	ADA	K6022	Anschluß von Geräten mit SIF 1000 im KME-Pegel 1 Eingabekanal, 1 Ausgabekanal
Anschlußsteuerung für Lochbandgeräte	ALB	K6025	Anschluß der Lochbandeinheit K6200 (SKR-Interface) 1 Eingabekanal, 1 Ausgabekanal
Anschlußsteuerung für Kassettenmagnetbandgerät	AKB	K5020	Anschluß von 2 Kassettenmagnetbandgeräten K5200 oder der Kassettenmagnetbandstation K5221
Anschlußsteuerung für Bildschirm	ABS	K7023	Anschluß des Bildschirmmonitors K7221
Anschlußsteuerung für Interface S 2	ASV	K8021	Anschluß von 2 duplexfähigen Kanälen nach CCITT-V24
Anschlußsteuerung für Tastatur und 2kanaliges serielles Interface	ATS	K7028	Anschluß einer Tastatur über gerätenahes Interface nach KROS-R5103 (Realisierung der Bedingungen des IFSS mit 20-mA-Stromschleife und Interface S2 nach CCITT-V24)
Anschlußsteuerung für EPROM-Programmiereinrichtung	PPE	K0420	Anschluß der EPROM-Aufnahmeeinrichtung PAE K0422 für die Programmierung des Schaltkreises U555
Anschlußsteuerung für die Bedieneinheit	ABD	K7022	Anschluß der Bedieneinheit BDE K2622 zur Inbetriebnahme, Wartung, Fehlersuche und Programmtestung

Tafel 8.1 (Fortsetzung)

Bezeichnung	Kurzbezeichnung	Chiffre	Hauptparameter
Anschlußsteuerung für Seriendrucker, Tastatur und alphanumerischer Kleinanzeige	ATD	K7026	Anschluß des Seriendruckers SD 1/SD 2 mit GNI, einer Einheitstastatur K7602 ... K7606 bzw. K7632 bis K7636 sowie einer alpha-numerischen Kleinanzeige ANK K7521
Busverstärker	BVE	K4120	Verstärkung und Entkopplung der Bussignale, ermöglicht die Verlängerung von System- und Daten-Koppelbus
Steckeinheiteneinsatz	STS		Baugruppeneinsatz zur Aufnahme von Steckeinheiten mit indirekten Steckverbindern, einbaufähig in Gehäuse des EGS, bis 480 mm aufreihbar Format der Steckeinheiten: 215×170 mm^2
		K0120	Einsatz für 5 Steckeinheiten, 120 mm breit
		K0121	Einsatz für 11 Steckeinheiten, 240 mm breit

statisch wie auch dynamisch arbeitende RAM-Schaltkreise Verwendung finden. Weitere Zusatzschaltkreise dienen zur logischen Anpassung der Systemschaltkreise untereinander und der Entkopplung der Anschlußeinheiten vom Bus und zum Anwender hin. Der Befehlssatz des Mikroprozessorschaltkreises U880 enthält 158 Basisbefehle und ist Grundlage der Assemblersprache SYPS K1520; er kann u.a. aus [8.27] [8.28] entnommen werden. Mit der Einheitsbaureihe der Stromversorgungsbaugruppen [8.29] steht ein Typensortiment von Schaltnetzteilen einschließlich Steuerbaugruppen zur Netz- und Zuschaltfolgesteuerung zur Verfügung, das den Aufbau von Stromversorgungseinheiten für beliebig konfigurierte Mikrorechner erlaubt.

In Tafel 8.1 sind die Komponenten des Mikrorechnersystems K1520 zusammengestellt. Für industrielle Steuerungssysteme werden im wesentlichen nur die zentrale Recheneinheit(en) und die Speicherkarten eingesetzt. Die Anschlußeinheiten für die datenverarbeitungstypischen Geräte werden relativ selten verwendet. Sofern sie in einer in den Folgeabschnitten beschriebenen Steuereinrichtung Anwendung finden, wird an diesen Stellen näher auf sie eingegangen.

Die zentrale Recheneinheit (ZVE) stellt die wichtigste Baugruppe im Mikrorechnersystem K1520 dar. Sie kann als Einplatinenrechner (außer K2525) betrieben oder als zentrale Steckeinheit an den K1520-Bus angeschlossen werden. In ihrer Grundvariante enthält sie den Prozessorschaltkreis, einen Schaltkreis zur parallelen Ein-/Ausgabe und einen Zähler-Zeitgeber-Baustein. Neben Zusatzschaltkreisen, z.B. zur Busentkopplung, befinden sich Schreib-Lese-Speicher (statisch RAM) mit einer Speicherkapazität von ein KByte und drei Steckfassungen zur Aufnahme von programmierbaren Festwertspeichern U555 auf dieser Karte. Ein quarzstabilisierter Taktgenerator sorgt für einen Systemtaktzyklus von 407 ns, während die Rücksetzschaltung den Programmstart nach Zuschalten der Versorgungsspannungen ab der Initialadresse ØØØØ H gewährleistet. Die Arbeitsweise der ZVE wird durch die in den Speichern abgelegten Befehlsfolgen bestimmt. Ein im Prozessorschaltkreis befindlicher Zähler (Programmzähler/Befehlszähler) ruft, von der Initialadresse beginnend, in stetiger Folge die vom Programmierer festgelegten und sich im Speicher befindlichen Befehle und Daten ab. Die Befehle werden vom Prozessor als Anweisungen über sein Verhalten (Befehlsabarbeitung) interpretiert, während die Daten zur eigentlichen Auslösung von Steuerreaktionen herangezogen werden. Nähere und ins

Detail gehende Erläuterungen über die Arbeitsweise von Mikrorechnern können aus [8.27] entnommen werden.

Der relativ geringe Speicherumfang auf der zentralen Recheneinheit erfordert in den meisten Fällen eine Speicherraumerweiterung. Dazu dienen die Baugruppen Festwertspeicher (PFS), Operativ-Festwert-Speicher (OFS) und Operativspeicher (OPS). Die Kartenbaugruppe PFS erlaubt die Erweiterung bis zu 16 BKyte EPROM U555, während die Karte OFS eine Kombination von 2 KByte statisch RAM und bis zu 6 KByte EPROM darstellt. Die Operativspeicherleiterkarten gestatten eine Erweiterung um 1 KByte (dynamisch RAM) und 4 KByte (statisch RAM). Bei der letzteren gibt es eine Bestückungsvariante mit CMOS-Speichern, die aufgrund ihres geringen Energieverbrauchs durch Batteriestützung auch bei abgeschalteter Anlage ($\geqq$ 200 h) ihren Inhalt nicht verlieren.

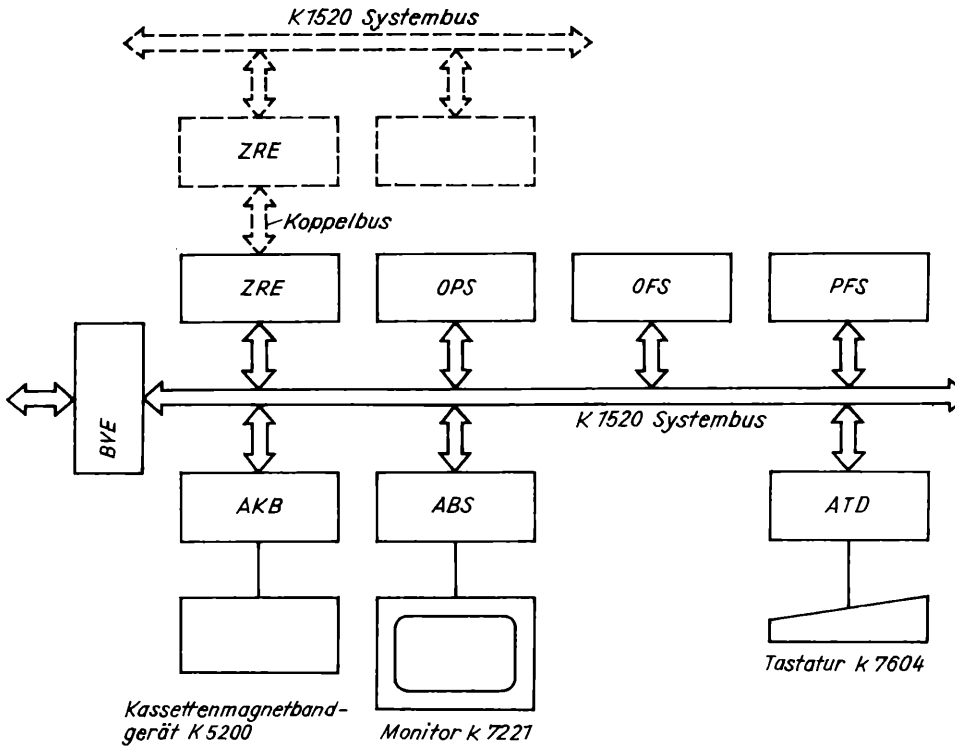


Bild 8.1. Beispiel einer Mikrorechnerkonfiguration K1520

Bild 8.1 zeigt an einem Beispiel den Anschluß der Komponenten des Mikrorechners an seinen Bus. Der K1520-Bus wird durch zwei Bündel von Leitungen gebildet. In der ersten Gruppe sind alle die für den Systemaufbau wesentlichen Leitungen einschließlich der der Stromversorgung zusammengefaßt. Sie dienen zur Kommunikation der ZVE mit den Speicher- und Eingabe-Ausgabe-Baugruppen und werden als Systembus bezeichnet. Der Koppelbus stellt die zweite Gruppe dar und dient u.a. der Kopplung mehrerer Mikrorechner K1520. Konstruktiv sind jeder Steckeinheit zwei 58polige Steckverbinder zugeordnet, wobei eine Steckverbindung in die gedruckte Rückverdrahtung (Systembus) eingebunden ist, während die andere durch Wickelverbindungen zu beschalten ist.

8.3. System 600

8.3.1. Übersicht

Das System 600 des VEB Numerik „Karl Marx" wurde in Zusammenarbeit mit dem Forschungszentrum des Werkzeugmaschinenbaus und den Werkzeugmaschinenkombinaten der DDR entwickelt. Es stellt ein leistungsfähiges mikroelektronisches Steuerungssystem dar, mit dem ein breites Spektrum von Steuerungsaufgaben gelöst werden kann. Das Grundkonzept beruht auf modularen Steckeinheiten, aus denen unterschiedliche Steuerungstypen konfiguriert werden. Bild 8.2 gibt einen Überblick über die gefertigten Steuerungstypen des Systems 600. Danach werden

- numerische Steuerungen (NC), die aufgrund der verwendeten Mikrorechnerbasis vom Typ CNC (*c*omputerized *n*umerical *c*ontrol) sind;
- programmierbare Maschinensteuerungen (PC *p*rogrammable *c*ontrol);
- Industrierobotersteuerungen (IRS)

unterschieden.

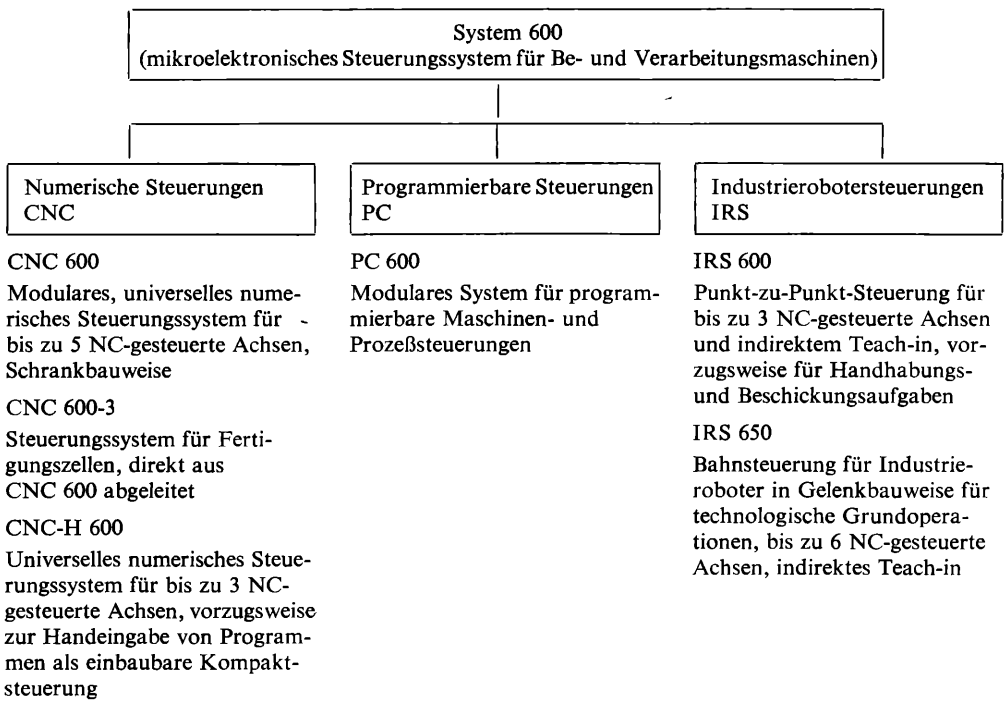


Bild 8.2. Übersicht über die Steuerungstypen des Systems 600

Steuerungen vom Typ CNC 600 werden vorzugsweise für Dreh-, Bohr- und Fräsmaschinen und zur Steuerung von Bearbeitungszentren eingesetzt (Bild 8.3).

Die CNC-H 600 ist als Kompaktsteuerung ausgelegt und in das Hängetabelau der (Bohr- oder Fräs-) Maschine oder ins Maschinenverdeck der (Dreh-) Maschine einbaubar (Bild 8.4). Die Steuerung ist für eine Handeingabe konzipiert, wobei besonderer Wert auf eine prozeßnahe, dialogunterstützte Programmierung vor Ort gelegt wurde. Die CNC-H 600 ist für maximal drei NC-gesteuerte Achsen ausgelegt und wird in drei Va-

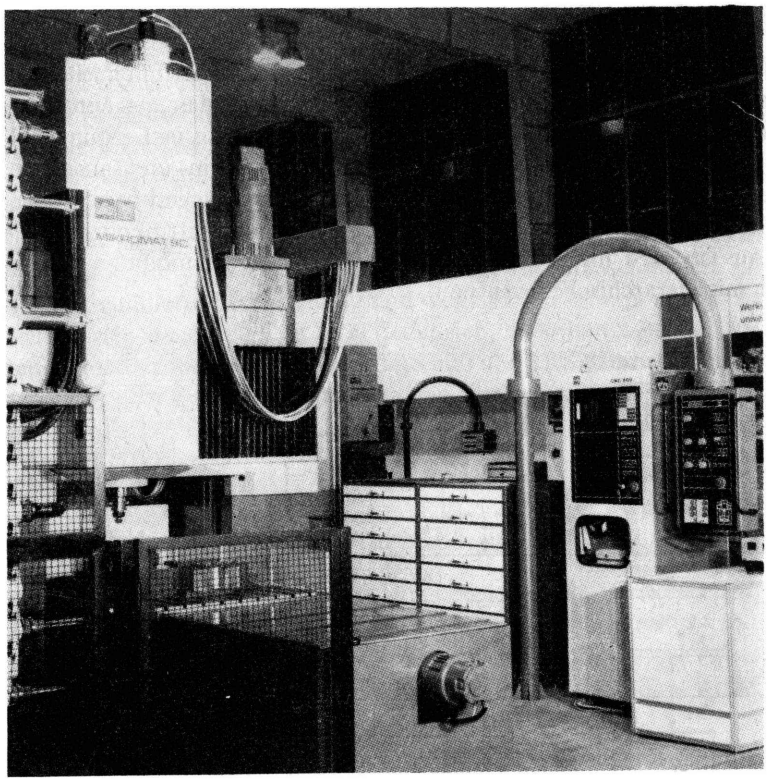


Bild 8.3
DNC 600 an einer Werkzeugmaschine (Werkfoto VEB NKM)

Bild 8.4. Gesamtansicht CNC-H 600 (Werkfoto VEB NKM)

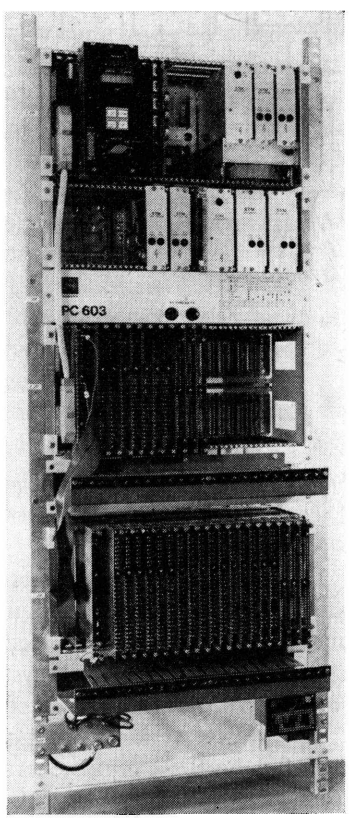


Bild 8.5. Gesamtansicht PC 603 (Werkfoto VEB NKM)

rianten hergestellt, die sich bezüglich der vorgesehenen Interpolationsart unterscheiden. Die speicherprogrammierbaren Steuerungen vom Typ PC 600 werden als autonome konstruktive und funktionelle Einheiten mit eigener Stromversorgung in Leistungsteilschränke eingebaut (Bild 8.5). Die Grundvariante umfaßt 480 Prozeßein- und -ausgänge. Die Industrierobotersteuerungen IRS 600 und IRS 650 werden nachfolgend beschrieben.

Alle Steuerungen des Systems 600 beruhen auf einem einheitlichen Systemkonzept. Die entsprechende Struktur ist dazu im Bild 8.5 dargestellt. Sie ist eine modulare Mehrmikrorechnerstruktur mit hierarchischem Aufbau.

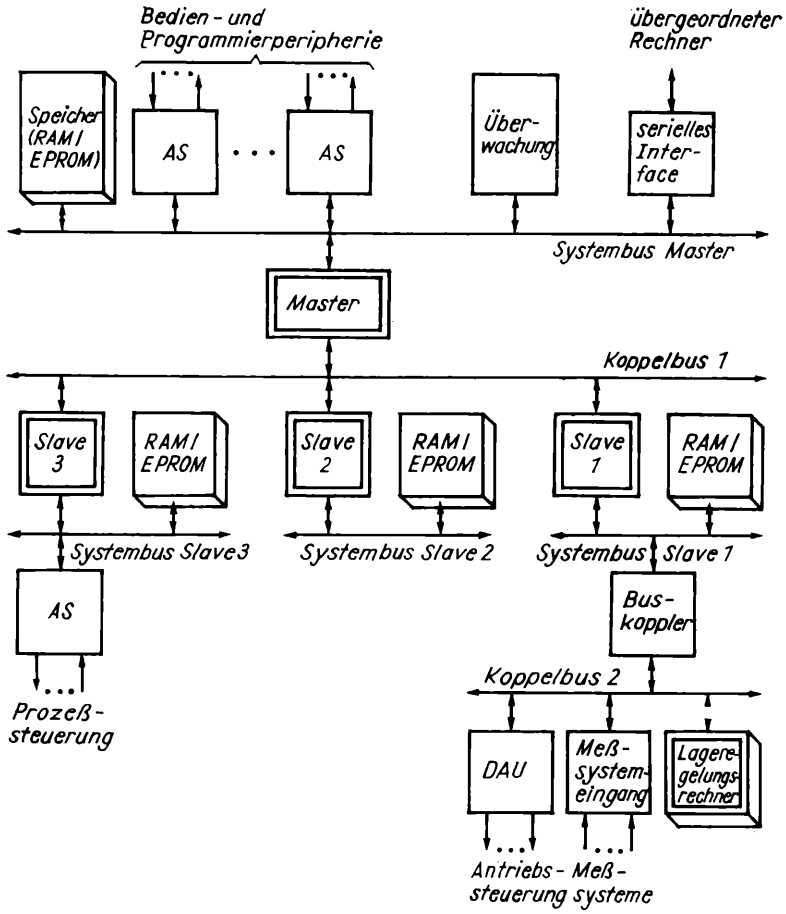


Bild 8.6. Zum Grundkonzept System 600
AS Anschlußsteuerung; *DAU* Digital-Analog-Umsetzer

Die Funktionen des Masters (Leitrechner) bestehen in der Koordinierung und Überwachung des Gesamtsystems und der Kommunikation mit dem Bediener/Programmierer und dem übergeordneten Rechner. Dazu sind am Systembus des Masters neben dem Speicher, der das Betriebssystem enthält und gleichzeitig operativer Speicher für die Anwenderprogramme ist, geeignete Anschlußsteuerungen angeschlossen.

Der Master verkehrt über einen Koppelbus mit maximal drei untergeordneten Rechnern für die Bewegungssteuerung (Slave 1), für die Ausführung umfangreicherer numerischer Operationen (Slave 2) und für die Prozeßsteuerung (Slave 3).

Der Systembus von Slave 1 ist mit einem weiteren Bus gekoppelt, an den die Lageregelungsrechner für die servogeregelten Achsen und die zugeordneten Meßeingänge und Stellausgänge angeschlossen sind.

An den Systembus von Slave 3 ist eine Anschlußsteuerung zur Ein- und Ausgabe binärer Signale für die Prozeßsteuerung angekoppelt. Aufgrund der gewählten Architektur und eines weitgehend vereinheitlichten Baugruppensystems ist es möglich, die Hardware weitgehend auf die konkrete Steuerungsaufgabe zuzuschneiden.

Eine detaillierte Beschreibung und wesentlicher Komponenten des Betriebssystems am Beispiel der Steuerung CNC 600 findet sich im Anhang. Bei den nachfolgend beschriebenen Industrierobotersteuerungen IRS 600 und IRS 650 handelt es sich um eine Einrechner- bzw. eine Zweirechnervariante, bei denen die mittlere Hierarchieebene aus Bild 8.6 im wesentlichen entfällt und alle Komponenten an den Systembus des Masters angeschlossen sind.

8.3.2. Industrierobotersteuerung IRS 600

Die IRS 600 ist eine Industrierobotersteuerung, die vorwiegend für Handhabungs- und Beschickungsaufgaben geeignet ist. Sie ist eine Punkt-zu-Punkt-Steuerung mit maximal drei simultan NC-gesteuerten Achsen. Zusätzlich können zwei weitere Achsen als nockengesteuerte bzw. reine Schaltachsen betrieben werden, wobei eine davon stufenlos über D/A-Wandler-Ausgang ansteuerbar ist.

Die NC-gesteuerten Achsen sind lagegeregelt. Es können direkt hydraulische Antriebe mit einem Strom im Bereich ± 50 mA bzw. elektrische Antriebe mit einem Spannungssignal im Bereich ± 10 V angesteuert werden. Zur Positionserfassung sind phasenzyklische Meßsysteme (Inductosyn, Resolver) anschließbar. Die Meßsystemauflösung beträgt 0,01 mm bzw. 0,01°. Die Programmierung und Bedienung der Steuerung erfolgt über das am IRS-600-Schrank (600 mm × 600 mm × 1800 mm) einhängbare tragbare Teach-in-Tableau. Neben der Möglichkeit, Programme im indirekten Teach-in einzugeben, besteht auch die Möglichkeit der Programmeingabe mittels eines ansteckbaren Lochstreifenlesers (Typen FS 100 bzw. FS 330). Zur Lochbandausgabe kann ein Stanzer der Typen DT-105 S bzw. daro 1215 angeschlossen werden.

Die Steuerung IRS 600 ist entwickelt und erprobt für den Industrieroboter IR 2/S2. Das betrifft Betriebs- und Diagnosesoftware sowie das in den Steuerschrank eingebaute Leistungsteil. Für Anwenderprogramme wird ein Speicherraum von 5 KByte CMOSRAM mit einer Stützzeit von mindestens 200 h zur Verfügung gestellt.

Der universelle, modulare Charakter der Steuerung ermöglicht selbstverständlich die Steuerung anderer Roboter im Rahmen des beschriebenen Funktionsumfangs. Die maximale Anzahl von Prozeßeingangssignalen ist 136, von Prozeßausgangssignalen 96.

8.3.2.1. Gerätetechnik

Der innere Aufbau der Steuerung ist im Blockschaltbild (Bild 8.7) dargestellt. Zu wesentlichen Teilen konstituiert sich die Steuerung aus Baugruppen des Systems K1520 und universellen Baugruppen des Systems CNC 600. Die IRS 600 stellt ein Einrechnerkonzept dar. Zentraler Rechner ist die ZRE K2521, die den standardisierten K1520-Systembus bildet.

Als Speicher für residente Programme wird zusätzlich die Baugruppe K3820 benutzt. Zwei CMOS-Steckeinheiten K3521 stellen einen Bereich von 8 KByte (batteriegestützt) zur Verfügung (Anwenderprogramme, Unterprogramme, Generierdaten). Über die Bau-

gruppe DA 2, die 4 D/A-Wandler-Kanäle mit einer Breite von 8 Bit besitzt, werden die Antriebe der drei NC-gesteuerten Achsen sowie einer nicht NC-gesteuerten Achse angesteuert. Für die Direktansteuerung von Servoventilen wird ein Stromsignal im Bereich ± 50 mA bei einem Lastwiderstand von 50 Ω erzeugt. Die Meßsystemansteuerung und Analogsignalverarbeitung erfolgen im Modul M1 (bei Inductosyn) bzw. M2 (bei Resolver). Die digitale Meßwertverarbeitung einschließlich der Bildung der Busschnittstelle zur Abfrage durch die ZRE befinden sich auf der je Meßachse einmal eingesetzten Steckeinheit ME 1. Der Lageregelungsalgorithmus wird im Rechner ausgeführt.

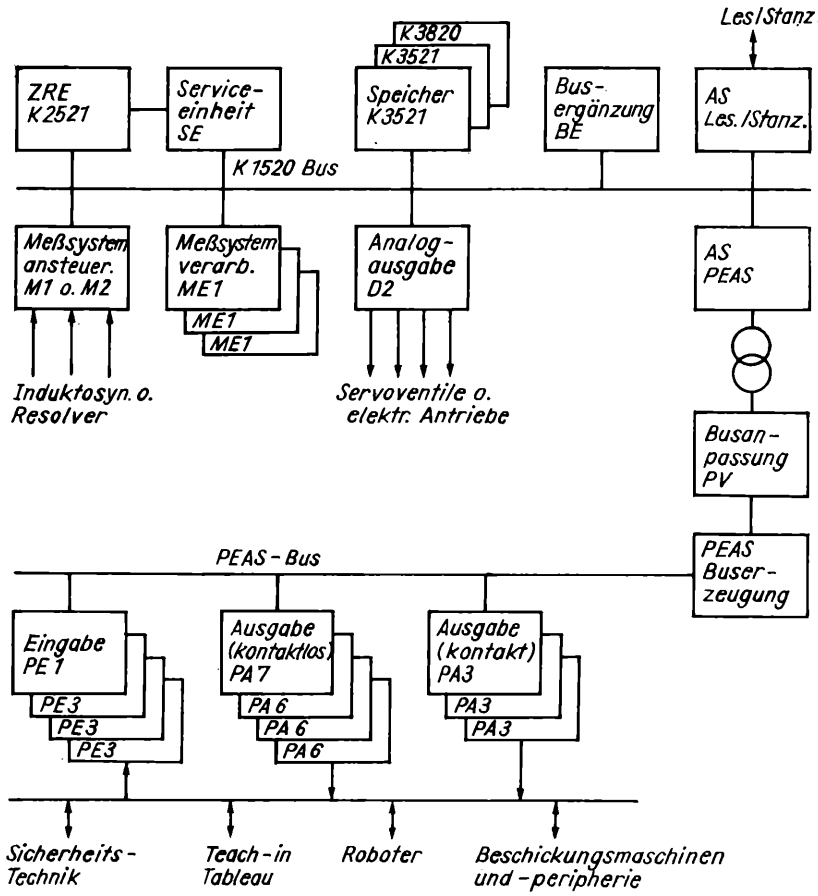


Bild 8.7. Blockschaltbild der IRS 600
statt D2 lies DA2

Mit der Kartenbaugruppe Serviceeinheit (SE), die fester Bestandteil jeder Steuerung ist und eigene Bedien- und Anzeigeelemente besitzt, können Diagnoseroutinen angewählt und gestartet, Prozeßsignale ausgegeben bzw. abgefragt, Daten und Adressen angezeigt sowie Achsdaten generiert werden. Die Serviceeinheit besitzt zu diesem Zweck vier Eingabetasten, einen Wahlschalter, eine 6stellige 7-Segment-LED-Anzeige sowie acht Lumineszenzdioden.

Die Steckeinheit Busergänzung hat spezifische Aufgaben, insbesondere Überwachungsaufgaben zu lösen. Mittels der Anschlußsteuerung AS PEAS erfolgt die Verbindung vom Rechnersystembus zum PEAS-Bus, wie dies bei den bereits behandelten Steuerungen dargelegt wurde.

Der Anschluß des Teach-in-Tableaus erfolgt am PEAS-Bus über einen Multiplexer der Datenbreite 8-Bit-Eingabe und 16-Bit-Ausgabe. Das Tableau (Bild 8.8) beinhaltet eine programmierbare Anzeige, eine numerische Eingabe, eine Funktionstastatur zur Befehlseingabe, Servicetasten, zwei Wahlschalter für Betriebsarten und Verfahrgeschwindigkeiten sowie Not-Aus- bzw. Ausschalter. Es ist konstruktiv im Schutzgrad IP 54 ausgeführt.

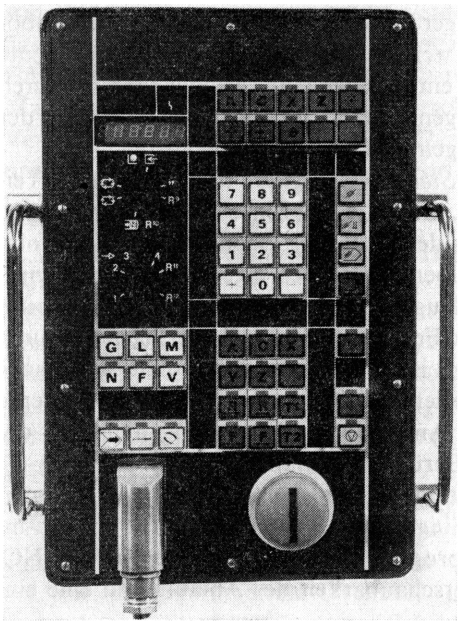


Bild 8.8. Teach-in-Tableau IRS 600 (Werkfoto VEB NKM)

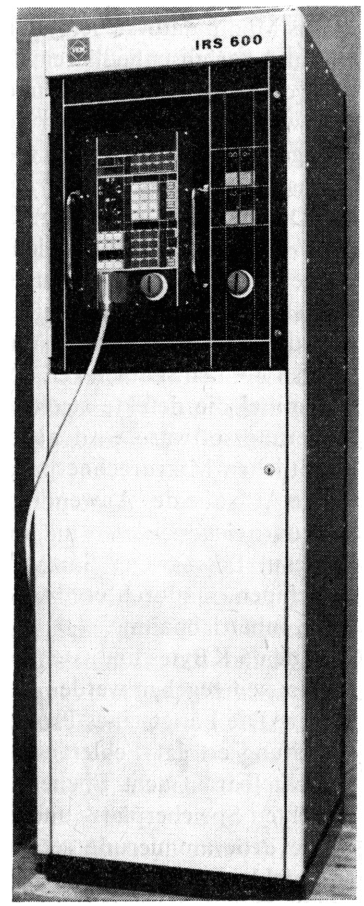


Bild 8.9. Steuerschrank IRS 600 mit Teach-in-Tableau (Werkfoto VEB NKM)

Am Schaltschrank selbst sind nur Hauptschalter, Taster für Steuerspannung, Hydraulik und Not-Aus-Schalter angebracht. Die Gesamtansicht der IRS 600 mit eingehängtem Teach-in-Tableau wird im Bild 8.9 gezeigt. Das Logikteil des Steuerschranks ist konstruktiv modular auf der Basis von in Baugruppenträgern steckbaren Einheiten, die über eine gedruckte Rückverdrahtung verbunden sind, aufgebaut. Die Stromversorgung erfolgt auf der Basis von Moduln des Systems K1510.

8.3.2.2. Programmsysteme und Programmierung

Die zur IRS 600 gehörenden Programmsysteme unterteilen sich in resident in der Steuerung auf EPROM abgelegte Programme und vom Anwender zu erstellende und auf CMOSRAM-Speicher abzulegende Ablaufprogramme.

Die residente Software besteht aus den beiden Hauptkomponenten Manipulatorsoftware und Steuerungsdiagnose.

Die *Manipulatorsoftware* setzt sich aus einem Betriebssystem und dem Manipulatorsteuersystem zusammen.

Hauptaufgaben des Betriebssystems sind dabei die Steuerung des Verkehrs mit der gesamten Eingabe-Ausgabe-Peripherie (Abfrage der Eingänge, Ausgabe an die Ausgänge, Bedienung von Leser und Stanzer sowie der Serviceeinheit), die Zeitorganisation, Unterprogrammbehandlung und Programmaufrufe.

Das Manipulatorsteuersystem organisiert die gerätespezifischen Funktionen des Roboters und der anschließbaren Zusatztechnik. Es wertet den Zustand der Bedienelemente des Teach-in-Tableaus aus und organisiert die entsprechenden Aktionen. Des weiteren überwacht es das Vorliegen von Fehlermeldungen bzw. nichtplausiblen Zuständen der Anlage und gibt entsprechende Fehlerkodierungen aus.

Das ebenfalls residente *Programmsystem zur Steuerungsdiagnose* besteht aus zwei Teilen. Der erste Teil ist die sog. Einschaltdiagnose, die bei jedem Einschalten der Steuerung durchlaufen wird und die die Hauptbaugruppen der Steuerung diagnostiziert. Ihr erfolgreicher Durchlauf ist Voraussetzung für den Übergang in den eigentlichen Steuerungsbetrieb. Der zweite Teil umfaßt die einzelnen Baugruppen zugeordneten Diagnoseroutinen, die, durch die Serviceeinheit aufrufbar, zur Fehlersuche im Off-line-Betrieb benutzt werden können. Dieser Teil des Diagnosesystems ist so konzipiert, daß ohne zusätzliche Meßmittel die defekte steckbare Einheit lokalisiert werden kann. Durch die residente Steuerungssoftware wird abgesichert, daß der Anwender der Steuerung nicht mit der eigentlichen Mikrorechnerprogrammierung in Berührung kommt.

Die Aufgabe des Anwenders reduziert sich damit auf die Eingabe von *Ablaufprogrammen* und *Generierdaten* mit den spezifischen Eingabemitteln der Steuerung (Teach-in-Tableau, Leser). Der Satzaufbau der Ablaufprogramme ist ähnlich denen von NC-Maschinen. Dadurch ergibt sich eine gute Überschaubarkeit des Ablaufs und eine einfache Inbetriebnahme.

In den 5 KBytes umfassenden, gestützten RAM-Speicher können bis zu 99 Ablaufprogramme eingegeben werden. Jedes Ablaufprogramm ist eine Folge von maximal 99 NC-Sätzen. Die Länge eines Programms darf jedoch 1 KByte nicht überschreiten (bei Überschreitung erfolgt Fehleranzeige). Die bis zu 99 adressierbaren Unterprogramme sind schachtelbar in acht Ebenen. Ihre Anwendung empfiehlt sich insbesondere unter den Aspekten Speicherplatz- und Programmierzeitoptimierung.

Die Programmierung setzt eine Generierung der G-Befehle und der Variablen der M-Befehle voraus.

Alle Details bezüglich der Programmerstellung und Generierung, der Korrektur von Programmen und Daten sowie der notwendigen Bedienhandlungen sind in der Bedienanweisung des IRS 600 fixiert. Nachfolgend soll deshalb nur auf einige ausgewählte Probleme eingegangen werden.

Die Teach-in-Programmierung kann auf drei verschiedene Arten erfolgen; dabei ist eine Kombination dieser innerhalb eines Programms möglich. Es wird zunächst unterschieden in die Teach-in-Programmierung ohne Ausführung von Achsbewegungen und in eine solche mit Ausführung von Achsbewegungen. Die erstgenannte Art entspricht einer Handeingabe aller Satzinformationen. Bei der Programmierung mit Achsbewegungen gibt es eine erste Variante, bei der alle Satzinformationen von Hand eingegeben werden, aber die Eingabe des Folgesatzes erst erfolgt, nachdem die Achsbewegungen des gerade eingegebenen Satzes ausgeführt und die Satzinformation in den Programmspeicher übernommen wurden. Bei der zweiten Variante werden keine geometrischen Informationen von Hand eingegeben. Die Achsen werden von Hand in die gewünschten Sollpositionen gefahren, und danach erfolgt deren Übernahme ins Programm.

Außerdem soll die Generierung von M- und G-Befehlen erwähnt werden, die bei einfacher Programmierung eine hohe Flexibilität der IRS 600 gewährleistet.

Mittels G-Befehlen können Zeitglieder, Koinzidenzbereiche, Koordinatentransformationen und Korrekturwerte programmiert werden. Durch Generierung der den M-Befehlen zugeordneten Variablen (insgesamt stehen 84 solcher Variablen zur Verfügung) können Verriegelungsbedingungen in das Routineprogramm eingefügt, Eingänge abgefragt, Ausgänge gesetzt oder rückgesetzt werden. Während die genannten Generierungen über das Teach-in-Tableau erfolgen, können mittels Serviceeinheit für Achsdaten, die vom Hersteller der Steuerung festgelegt werden, neue Werte generiert werden. Das betrifft Maximalgeschwindigkeit, Einrichtegeschwindigkeit, Koinzidenzbereiche, Einfahrkurve, Verfahrbereich, Nockenpositionen. Damit wird die Möglichkeit gegeben, die Steuerung optimal an die jeweiligen Anwenderforderungen anzupassen.

8.3.3. Industrierobotersteuerung IRS 650

Die IRS 650 ist eine Industrierobotersteuerung, die als freiprogrammierbare Bahnsteuerung für prozeßflexible Industrieroboter für technologische Operationen und als Weiterentwicklung auch für Beschickungsaufgaben (als IRS 651) geeignet ist. Sie ist für Industrieroboter mit bis zu sechs vorwiegend elektrisch angetriebenen NC-Achsen ausgelegt, deren zeitlich miteinander koordinierte Bewegungen zur Erzeugung räumlicher Kurven gesteuert werden.

Als erste Einsatzfälle wurde die IRS 650 an die Gelenkroboter IR 10 und 60 des VEB Kombinat „Fortschritt" Landmaschinenbau angepaßt, auch in Verbindung mit Schweißeinrichtungen des Zentralinstituts für Schweißtechnik Halle. Die Achsen sind lagegeregelt mit unterlagertem Drehzahl- und Stromregelkreis. Zum Einsatz gelangen die drehzahlgeregelten Thyristorantriebe AA 100 mit 2pulsiger Schaltung. Sie wurden speziell für das Roboterprogramm der DDR entwickelt; die Gleichstrommotoren RSM 10 (1,3 Nm) und RSM 60 (3,5 Nm) sind dabei in Kurzbauweise roboterfreundlich ausgelegt. Für ausreichende Dynamik ist immer eine genügend große Drehmomentreserve vorzusehen. Die sechs Achsen sind in zwei Gruppen einzuteilen. Drei Hauptachsen für die Orientierung des Greifers im Raum, drei Nebenachsen für die zusätzliche Feinorientierung des Greifer-Werkzeug-Systems.

Für die Lageerfassung dienen inkrementale Geber IGR 400 mit Nullimpuls (VEB Carl Zeiss, JENA), die 400 Impulse/Umdrehung erzeugen, was eine Auflösung kleiner als 1° an der Motorwelle bedeutet. Sie werden direkt an die Antriebsmotoren angeflanscht. Die Signalübertragung ist mittels Kabels auf 15 m Entfernung möglich. Die Lageregelkreise sind digital mittels Mikrorechners ausgeführt und geben die Sollwerte im Bereich ± 10 V an den unterlagerten Drehzahlregelkreis aus. Zur Umwandlung der digitalen Werte in analoge Spannungen dienen 14-bit-D/A-Wandler.

Erreichbare Hauptparameter der Bewegungsvorgänge sind

- programmierbarer Vorschub 18 bis $20 \cdot 10^3$ °/s an der Motorwelle mit einer Handbeeinflussung von 5 bis 100 % (Override);
- maximale Geschwindigkeit 2 m/s bei einer umgerechneten Auflösung von 0,1 mm, Motordrehzahl 3000 U/min;
- optimale Beschleunigung einstellbar;
- Schrittmaß 1 K, 10 K oder 100 K mal Wegauflösung (K generierbare Konstante);
- maximale Weglänge $8 \cdot 10^6$ mal Wegauflösung;

- Referenzpunkt, Softwareendschalter, Sicherheitsendschalter als öffnender Schalter, Näherungsinitiatoren.

Programmiert wird die Geschwindigkeit der Leitachse, d.h. der Achse mit dem größten Weg zwischen zwei Punkten P_1 und P_2. Die Steuerung beaufschlagt die anderen Achsen so, daß P_2 von allen Achsen gleichzeitig erreicht wird.

Die IRS 650 kann extern textuell programmiert werden, wobei die Programmeingabe mittels Magnetbands (KMBG K5200) erfolgt. Vorzugsweise wird das Bearbeitungsprogramm aber im indirekten Teach-in-Verfahren vor Ort erstellt. Die Achsen werden dabei von Hand verfahren, und nach Erreichen der gewünschten Positionswerte werden diese in den Programmspeicher übernommen, während die Daten des übrigen Satzaufbaus (nach TGL 200-0863), wie Programm- und Satznummer, Weg- und Hilfsbedingungen, Unterprogramme, Geschwindigkeiten sowie Variable, über Tastatur eingegeben werden können. Wie bei NC-Programmen sind Programmverzweigungen, -schleifen und -sprünge, Zähl- und Zeitstufen möglich; spezifisch für Roboter sind Stapelalgorithmen installiert.

Für den Anschluß an einen übergeordneten Rechner (DNC) zur Produktionssteuerung und -datenerfassung sowie von Peripherie sind standardisierte Interfaces V 24 (RS 232) bzw. IFSS als bidirektionale Datenkanäle vorgesehen.

Inbetriebnahme und Service werden durch Einsatz des Bildschirmgeräts K8911 (VEB Kombinat Robotron) mit komfortablem Dialog unterstützt.

Zur Verbindung mit dem Prozeß stehen 48 digitale Eingänge einschließlich Sensoreingänge, mit 10 bis 20 mA belastbar, und 48 digitale Ausgänge, mit 100 mA belastbar, zur Verfügung, die neben Notfunktionen, Endschalterinformationen, Start, Stop usw. ausreichend wahlfreie Belegungen gestatten. Als einheitlicher Signalpegel am Steuerungssystem wurden 24 V GS gewählt. Höhere Stromstärken und Wechselspannung bedingen das Zwischenschalten eines Relais, z.B. GBR 20.1.

Die gewünschte Betriebsart ist am Schrank einstellbar, wobei ihre Veränderung mit der Umschalthandlung wirksam wird, wenn diese zulässig ist.

Vorgesehen sind:

Handbetrieb, Einrichten

Achsen bewegbar, Referenzpunkt nicht notwendig anzufahren, reduzierte Geschwindigkeit (wird angezeigt).

Referenzpunkt

Einzeln je Achse von Hand oder selbständige Aufnahme des Referenzpunktes, Ausführung wird angezeigt.

Programmieren

Teach-in-Tableau aktiv, Übernahme in den Programmspeicher.

Automatik

nur nach Referenzpunktaufnahme wirksam.

Alle vorhandenen Programme startbar; nach Stop Übergang in ein anderes Programm möglich.

Programmein/-ausgabe

Übernahme des Bearbeitungsprogramms vom Halbleiterspeicher auf die Magnetbandkassette und umgekehrt.

8.3.3.1. Gerätetechnik

Die IRS 650 ist in einem Schrank 600 mm × 600 mm × 1800 mm untergebracht, der die Logik, die PC, die Leistungsglieder und die Einphasen-Thyristorstromrichter TDR 100 einschließlich Drosseln aufnimmt. Der Schutzgrad IP 54 wird durch Zweikreiskühlung (Wärmetauscher) erreicht. Das Magnetbandkassettengerät (Schutzgrad IP 43) und das Teach-in-Tableau (Schutzgrad IP 54) sind extern angeordnet. Die Hardwarestruktur der IRS 650 ist im Bild 8.10 dargestellt.

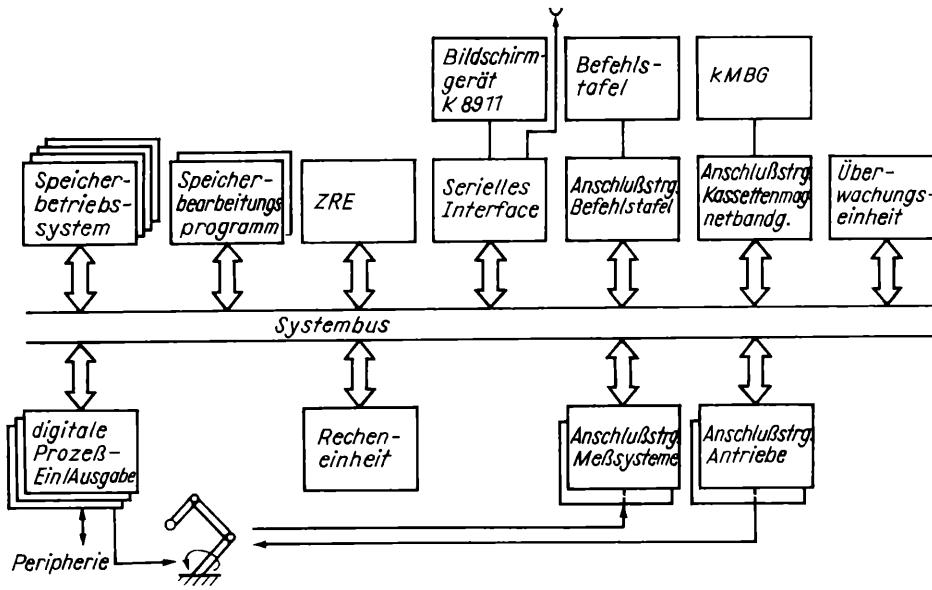


Bild 8.10. Blockschaltbild der IRS 650

Die Steueralgorithmen werden in Mikrorechnern verarbeitet, wobei deren Basis der Mikroprozessor U880 ist. Die Logikkassette koppelt mittels gedruckter Rückverdrahtung über den K1520-Systembus (TGL 37271/01) die beiden Recheneinheiten (ZRE 2521 des K1520 und die speichergekoppelte Recheneinheit) mit den Speichern, den Anschlußsteuerungen und den Ein-/Ausgabe-Modulen. Fest installiert für das Betriebssystem sind 32 KByte EPROM und 8 KByte CMOS-RAM.

Vorgesehen sind maximal 20 KByte für interne Anwenderprogramme, wahlweise 16 KByte gestützter CMOS-RAM oder 4 KByte gestützte CMOS-RAM und 16 KByte DRAM (bzw. EPROM). Damit sind bis zu 800 Raumpunkte speicherbar. Eine EPROM-Programmiereinheit (K0420) für die Datenübernahme RAM-EPROM bzw. KMB-EPROM (mit Bildschirmgerät K8911) ist anschließbar.

In den Anschlußsteuerungen werden die peripheren Signale in die busadäquate Form gewandelt. Das KMB-Gerät wird interruptgesteuert angeschlossen. Anzeigefunktionen erfüllen eine 4stellige 7-Segment-Anzeige und 16 LED auf dem Teach-in-Tableau. Zur Überwachung der Stromversorgung und der Temperatur sind Vorkehrungen getroffen, und eine spezielle KBG UEW dient der Überwachung rechnerinterner Funktionen.

Bei Fehlern werden neben der Sicherheitsschaltung Datenrettungsroutinen gestartet.

Die Signale der Prozeß-Ein-/Ausgabe werden angezeigt, um die Signalverläufe verfolgen zu können.

Die Ausgabe von Informationen an den Prozeß wird erst nach fehlerfreier Grunddiagnose und damit Betriebsbereitschaft freigegeben.

Die Mikrorechentechnik erlaubt die Eigendiagnose der gesamten Hardware. Das entsprechende Programm wird über Magnetband geladen; die Aussage erfolgt auf dem anschließbaren Bildschirmterminal K8911.

Die steuer- und regeltechnische Anpassung der im Betriebssystem enthaltenen Funktionen geschieht durch Generierung im Inbetriebnahmeprozeß.

Zur Anzeige gelangen wahlweise

- Satznummer;
- Haupt-, Unterprogrammnummer;
- Fehler (nach blinkender Statusanzeigetaste erfolgt die Anzeige durch Tasterbetätigung).

Die Zehnertastatur hat Doppelfunktionen, wobei automatisch umgeschaltet wird entsprechend der Syntax des Satzaufbaus.

8.3.3.2. Programmsysteme

In der IRS 650 sind zwei Arten von Programmen zu unterscheiden:

- residente Programme im EPROM der Steuerung, die jederzeit aufgerufen werden können;
- nachladbare Programme auf Magnetbandkassette, die nur zeitweilig benötigt und daher von Hand in die internen (RAM-)Speicher geladen werden.

Das zentrale residente Programm ist das *Satzbetriebssystem* SABE. Es ermöglicht:

- die Ein-/Ausgabe von Informationen in Verbindung mit der Befehlstafel;
- das Editieren des Bearbeitungsprogramms in der Programmier- bzw. Optimierungsphase;
- eine Bewegung der Roboterachsen durch Handbefehle, indem zeitweilig Wegsätze gebildet werden, die von der Wegsteuerung weiterverarbeitet, jedoch nicht dauerhaft abgespeichert werden;
- den Einzelsatzbetrieb des Bearbeitungsprogramms;
- die Übersetzung des Bearbeitungsprogramms (Anwenderprogramm) vom Quellkode nach TGL 200-0863 in den internen Objektkode

Das NC-Programmformat wurde wegen seiner allgemeinen Verbreitung zum Standardeingabeformat gewählt; es bedingt jedoch relativ viel Speicherplatz, und seine Verarbeitung ist zeitaufwendig.

Deshalb wird der Quellkode sofort umgesetzt in

- Wegsätze von 24 Byte, die alle Informationen für die Bewegung $P_n \rightarrow P_{n+1}$ enthalten und die von der Wegsteuerung weiterverarbeitet werden;
- PC-Sätze von 12 Byte, die in der PC oder von SABE selbst abgearbeitet werden.

Auch die Information auf dem Magnetband im Format nach ISO 3407 wird in den Objektkode während des Einlesens gewandelt, umgekehrt beim Archivieren.

Das Satzbetriebssystem übersetzt die im Bearbeitungsprogramm absolut stehenden geometrischen Daten in relative Maße. Das residente Programm *Wegesteuerung* (WST) errechnet die Sollwerte für die Lageregelkreise, indem die je Achse generierbaren Beschleunigungs- und Bremsvorgänge geführt sowie die Bewegung der bis zu sechs Achsen koordiniert werden. Durch WST wird vom Satzbetriebssystem die Folgeinformation abgerufen und zwischengepuffert.

Das residente Programm *programmierbare Steuerung* PC steuert die technologischen

Prozesse durch Interpretation Boolescher Gleichungen und Abfrage bzw. Beaufschlagung der Ein-/Ausgabe-Kanäle.

Der Lageregelungsalgorithmus wird vom Programm *Lageregelung* LAR abgearbeitet, wozu der zweite Mikrorechner der IRS 650 dient. Die Tastzeit für sechs Achsen beträgt 10 ms. Dieses Programm sorgt für die Referenzpunkterfassung, es überwacht die zulässigen Achsbewegungen und gestattet die Anpassung der Konstanten der Regelalgorithmen an die dynamischen Eigenschaften der Achsen durch Generierung.

Die Programme für *Ein- und Ausgabe* an die Befehlstafel, die Bildschirmeinheit und das Kassettenmagnetbandgerät können von den anderen Programmen genutzt werden.

Die nachladbaren Programme (off line) *Monitor* und *Editor* MON/EDIT bedingen das Vorhandensein der Bildschirmeinheit K8911. Sie erwirken die Anzeige und die Behandlung des Programms sowie von Einzeldaten und gestatten Veränderungen im Bearbeitungsprogramm. Diese Dienstprogramme steigern den Korrekturkomfort. Die *Einschaltprogramme* (On-line-Diagnose) stellen den Initialzustand her und realisieren die Einschaltdiagnose. Sie geben die Bearbeitung durch Betriebsbereitschaftsmeldung frei bzw. melden Fehler.

Die beiden nachladbaren Programme *Generierung* GEN und *Offline-Diagnostik* DIAG dienen der Inbetriebnahme des Industrieroboters durch Unterstützung bei der Eingabe von Generierdaten (z. B. Verstärkung, Drehsinn, Beschleunigung, Geschwindigkeit beim Einrichten, Softwareendschalter) und dem Service, d. h. der Fehlersuche bis zur Steckeinheit sowie dem zyklischen Test. Es lassen sich auch die residenten Programme kontrollieren. Die Steuerung gestattet zwei Arbeitsweisen:

- Echtzeitbetrieb (Antriebe aktiv)
- Testbetrieb (Antriebe inaktiv).

Zur Sicherung der Zeitbilanz werden einige Programmkomplexe als Hintergrundprogramme betrieben, die von den Echtzeitprogrammen mittels Interrupts, wenn nötig, unterbrochen werden.

8.3.3.3. Programmierung

Die geometrischen Daten werden im indirekten Teach-in programmiert, indem durch Handbetätigung der Achsantriebe die Position angefahren und durch Tastendruck das Speichern ausgelöst wird. Die Befehle im Wort werden durch Tasten eingegeben. Die Programmiervorschrift entspricht dem NC-Programm, TGL 200-0863 mit einem möglichen Befehlsvorrat:

N xxx; F xxxx; G xx; H xxxx; L xx; M xx; % xx;
Ausgänge HX0 xx; Eingänge HX1 xx; Zeitstufe H02 xx; Zähler H03 xx.

Die Geschwindigkeit F wird als Vielfaches der Winkelauflösung angegeben, 21 Zusatzfunktionen M sind definiert. Die Wegbedingungen G bezeichnen Punktsteuerung, Bahnsteuerung, Sensor- und Referenzfunktion, Fein- und Grobpunkt.

Wie allgemein üblich, ist eine Bedienerführung installiert, wobei LED die Adresse angeben und die Ziffern angezeigt werden. Bei Wortende blinken die zulässigen Adreßtasten für das Folgewort, bei Satzende blinkt die Taste Satzabschluß. Während Blinken zur Eingabe animiert, wird durch dauerndes Leuchten die Betätigung quittiert.

Extern kann mittels Bürorechners A5130 mit dem Programmpaket E IRS 650 ein Programm textuell auf eine Magnetbandkassette geschrieben werden, das vor Ort an der IRS 650 korrigiert werden kann.

Eine große Erleichterung tritt ein, wenn off-line alle PC-Sätze programmiert werden und vor Ort nur die Geometriesätze eingefügt werden.

8.4. Industrierobotersteuerung ursalog 5021

Die Industrierobotersteuerung ursalog 5021 des VEB EAW Berlin wurde in Zusammenarbeit des Instituts für Regelungstechnik Berlin und der Ingenieurhochschule Mittweida entwickelt und ist für die Steuerung von im direkten Teach-in programmierbaren und hydraulisch angetriebenen technologischen Industrierobotern vorgesehen. Als kinematische Grundeinheiten sind vor allem die für die Operationen Beschichten/Farbspritzen, Strahlen, Wischen und Putzen entwickelten Gelenkarmsysteme TR 79 und TR 10 des VEB „Sachsenring" Automobilwerke Zwickau vorgesehen. Entsprechend den genannten Einsatzgebieten leiten sich folgende die Struktur und den Aufbau der Steuerung wesentlich prägende Forderungen ab:

- Speicherung des Ort-Zeit-Verlaufs der zu programmierenden Bahnkurve im echten Zeitbezug und die Möglichkeit der Einflußnahme im Sinne einer Zeitraffung und Zeitdehnung bei der Wiedergabe (z. B. für die Synchronisation mit Fördersystemen);
- Einfügen bzw. Löschen von programmorganisatorischen und das Aktionsverhalten bestimmenden Befehlen nach erfolgter Programmierung der Bahnkurve;
- Einsatz von Positionsmeßsystemen und Antriebsstellelementen einschließlich deren Signalverarbeitungseinheiten, die den Forderungen nach Explosionsschutz entsprechend TGL 30020 gerecht werden.

Im Gegensatz zu verschiedenen Industrierobotersteuerungen für das direkte Teach-in-Programmierverfahren [8.23] [8.24], bei denen Massenspeicher mit seriellem Zugriff (z. B. Folienspeicher) als Arbeitsspeicher für die Aufzeichnung und für die Wiedergabe des Roboterprogramms Anwendung finden, werden bei der Steuereinrichtung ursalog 5021 Halbleiterspeicher dafür verwendet. Lediglich zur Archivierung von Roboterprogrammen findet ein Kassettenmagnetbandgerät Anwendung. Damit wird sowohl dem Trend nach Einsatz von Halbleiterspeichern mit immer größerem Speichervermögen als auch der Forderung nach Erhöhung der Zuverlässigkeit Rechnung getragen.

Die Industrierobotersteuerung ursalog 5021 ist unter Verwendung von Baugruppen des Mikrorechnersystems K1520 des Systems 600 und des industriellen Steuerungssystems ursalog 5020/ursadat 5000 konfiguriert; der funktionelle Inhalt wird weitgehend durch Software realisiert. Eine Beschreibung des Systems 5020 und insbesondere der in der Industrierobotersteuerung ursalog 5021 verwendeten Baugruppen befindet sich im Anhang 2.

8.4.1. Allgemeine Charakteristik

Die Industrierobotersteuerung ursalog 5021 ist als Multipunktsteuerung mit Interpolator konzipiert. Großer Wert wird auf einen möglichst breiten, nicht nur auf eine einzige technologische Operation zugeschnittenen Funktionsinhalt gelegt. Durch die beim Hersteller durchzuführende Grundinitialisierung ist die Steuereinrichtung in Verbindung mit geeigneten kinematischen Grundeinheiten für alle die technologischen Grundoperationen geeignet, bei denen der Bewegungsablauf bei der Programmierung zweckmäßig im direkten Zeitbezug vorgegeben wird. Die wesentlichsten Hauptparameter der Steuereinrichtung sind:

- statische Positioniergenauigkeit: $\pm 0{,}2$ mm bzw. $\pm 0{,}2°$ bezogen auf Antriebselemente mit einem Hub von 220 mm bzw. einer Drehung von 210° (die auf das Werkzeug bezogene Positioniergenauigkeit ist von der Bauform der kinematischen Gelenkeinheit und deren momentane Position im Raum abhängig);

- Anzahl der lageregelbaren Achsen: maximal 8, abrüstbar auf 6, 4, 2 Achsen;
- Auflösung des Stellwegs bei potentiometrischen Meßsystemen: 12 Bit;
- Programmdauer: Die maximale Programmdauer ist von der Anzahl der betriebenen Achsen, dem vorgewählten Faktor für den Taktteiler und der Ausrüstungsvariante bezüglich des Arbeitsspeichers abhängig. Für die vollständige Speicherausrüstung mit 256 KByte sind in der Tafel 8.2 die möglichen Programmlaufzeiten angegeben;

Tafel 8.2. Mögliche Programmdauer in Abhängigkeit der Achsanzahl und des Taktteilers bei einer Speicherausrüstung von 256 KByte

Achsen	Taktteiler 1	2	4	8	16
2	320 s	640 s	1280 s	2560 s	5120 s
4	210 s	420 s	840 s	1680 s	3360 s
6	160 s	320 s	640 s	1280 s	2560 s
8	130 s	260 s	520 s	1040 s	2080 s

- Interpolator: Glättungsinterpolator 3. Grades mit extern einstellbarem Taktteiler;
- Anzahl der aufrufbaren Programme: 128;
- Ausgabe des Stellsignals:
  für elektrische Antriebe ± 10 V, für hydraulische Antriebe ± 50 mA (Lastwiderstand 50 Ω);
- Anzahl der binären Ein- und Ausgänge:

  32 Eingänge über Optokoppler I_E 8,6 bis 12,2 mA, U_E 12 V $\pm$ 10% } bei 1-Signal
  I_E 0,5 mA, U_E 1,4 V } bei 0-Signal

  24 Ausgänge potentialgetrennt $U_{A\,max} = 60$ V Gs/Ws
  $I_{A\,max} = 0{,}5$ A
  $P_{max} = 10$ W

  Eine beliebige Anzahl von Eingängen und Ausgängen können für einen bestimmten Anwendungsfall durch eine Anwenderinitialisierung maskiert werden und stehen danach nicht im Sinne einer freien Programmierbarkeit unter Verwendung des Befehlssatzes zur Verfügung;
- Befehlssatz der Steuereinrichtung: 21 Befehle (Synchronisations-, Verzweigungs-, Verzögerungs- und Prozeß-Ein-/Ausgabe-Befehle);
- interner/externer Takt: wahlweise Synchronisation auf internen oder externen Takt. Durch Bereitstellung eines externen Taktes kann die Bewegung des Roboters an periphere Einrichtungen angepaßt werden (z. B. Förderbandsynchronisation). Bei externer Taktvorgabe ist eine Zeitraffung bzw. Zeitdehnung um ± 10% gegenüber der Aufnahme möglich;
- sekundärer Massenspeicher für Programmarchivierung:
  Kassettenmagnetbandgerät K5200;
- Bedienung und Anzeige: Die Bedienung und Anzeige erfolgt über eine alphanumerische Tastatur und eine alphanumerische Anzeige. Zur Vermeidung von Bedienfehlern ist die weitestgehende Führung des Bedieners realisiert (Menütechnik). Die Tastatur mit Anzeige befindet sich im Steuerschrank; sie kann bei Programmierungsarbeiten vom Schrank abgesetzt betrieben werden.

- Diagnosefunktionen: Zur Vermeidung von Havarien laufen nach dem Einschalten und während des Betriebs der Steuereinrichtung komplexe Diagnoseprogramme ab. Zur Überprüfung der Funktionsfähigkeit des Gesamtsystems sind eine Reihe von durch den Bediener aufrufbaren Serviceprogrammen implementiert. Zu Testzwecken kann durch Bahnvorgabe über das Kassettenmagnetbandgerät das definierte Verfahren des Roboters erzielt werden.
- Explosionsschutz: Auf Wunsch wird die Steuereinrichtung für das Betreiben des kinematischen Systems in explosionsgefährdeten Räumen (Gasexplosionsgefährdungsgruppe EG 3, Zündgruppe T2, Zündwilligkeitsgruppe II B) für potentiometrische Positionsmeßsysteme (Eigensicherheit) geliefert. Für die Stellsignalausgabe (hydraulische Antriebe) ist die Schutzart „Sonderschutz" durch das Servoventil zu realisieren. Die binären Eingangs- und Ausgangssignale sind nicht explosionsgeschützt; für den Ex-Schutz sind entsprechende pneumatisch-elektrische Signalwandler vorzusehen (z. B. PE-Wandlerblock PE 1B).
- Gefäßsystem: EGS-Schrank mit Sichtfenster 800 mm × 1600 mm × 600 mm, Schutzgrad IP 54.

8.4.2. Bedienung und Programmierung

Die Robotersteuerung ursalog 5021 ist eine freiprogrammierbare Steuereinrichtung für Bewegungs- und Aktionsabläufe mit eigenem Befehlssatz. Für die Bedienung und Programmierung steht eine alphanumerische Tastatur als Eingabegerät und eine 32stellige 5 × 7-Punktmatrix-Anzeige für die Darstellung von Texten zur Verfügung. Bis auf wenige, direkt in das Starkstromteil eingreifende Bedienelemente (z. B. Not-Aus) erfolgen alle Bedienhandlungen über diese Tastatur. Konstruktiv bildet die Tastatur mit der Anzeige eine Einheit. Zur Gewährleistung des Schutzgrads IP 54 ist diese innerhalb des Steuerschranks angeordnet und durch Öffnen einer mit einem Fenster versehenen Tür zugänglich. In der Regel wird die Bedienung/Programmierung von da aus erfolgen; es besteht jedoch die Möglichkeit, die Tastatur mit der Anzeige aus ihrer Halterung zu entfernen und sie vom Steuerschrank abgesetzt zu betreiben.

Das Bedienkonzept der Steuereinrichtung baut auf drei Ebenen auf; sie unterscheiden sich durch die an den Bedienenden zu stellenden Anforderungen:

- Bedienung im Sinne der Herstellung der Wiedergabebereitschaft bei vorhandenen Roboterprogrammen auf Magnetbandkassette;
- Programmierung von neuen Bewegungs- und Aktionsfolgen und deren Archivierung auf Magnetbandkassette;
- Durchführung von Testoperationen zur komplexen Überprüfung/Diagnose des Gesamtsystems, bestehend aus dem kinematischen Grundgerät, den peripheren Einrichtungen und der Steuerung.

Während die erste Stufe der Bedienung durch einen mit den wesentlichsten Bedienhandlungen vertrauten Operator vorgesehen ist, erfordert die zweite Stufe Kenntnisse über den Befehlssatz der Steuerung und das Verhalten peripherer Komponenten (z. B. Teileerkennungssystem). Die höchste Stufe ist für versierte Servicetechniker vorbehalten und erlaubt die komplexe Bedienung im Rahmen von Erstinbetriebnahmen oder bei Fehlersuche.

Im Bild 8.11 wird die der Bedienerführung zugrunde gelegte Baumstruktur gezeigt. Nach dem Einschalten der Steuereinrichtung wird dem Bedienenden mit der Anzeige

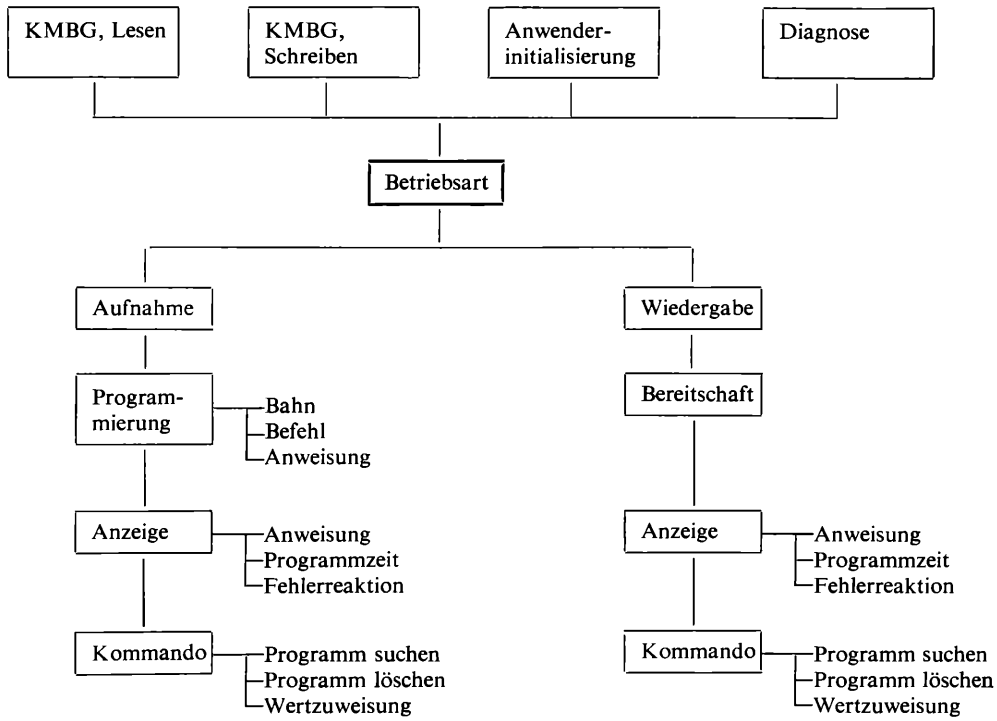


Bild 8.11. Baumstruktur der Bedienerführung

„Betriebsart" signalisiert, daß er eine der möglichen Betriebsarten „Aufnahme", „Wiedergabe", „Kassettenmagnetbandgerät lesen", „Kassettenmagnetbandgerät schreiben", „Anwenderinitialisierung" oder „Diagnose" auswählen muß. Mit der Betätigung einer Quittungstaste wird ihm die Betriebsart „Aufnahme" angeboten. Soll eine Programmaufnahme oder -modifizierung erfolgen, so wird in diesem Ast der Baumstruktur verblieben und dem Operator die „Programmierung" durch Anzeige angeboten. Wünscht er ein Programm zu löschen, gelangt er durch zweimaliges Betätigen der Organisationstaste [→] zur Eingabemöglichkeit eines „Kommandos". Nach erfolgter Quittierung wird dem Bedienenden zuerst die Möglichkeit der Suche nach einem im Speicher befindlichen Programme angeboten. Da er jedoch ein Programm löschen möchte, betätigt er erneut die Organisationstaste [→] und bekommt mit „Programm löschen" die gewünschte Betriebsart angeboten. Sobald die Quittung erfolgt, wird er durch Anzeige aufgefordert, die Nummer des zu löschenden Programms einzugeben. Nach Betätigung der Quittungstaste wird das Programm gelöscht und erneut „Programm löschen" angeboten. War nur ein Programm zu tilgen und soll im weiteren die Betriebsart „Wiedergabe" aufgerufen werden, so wird durch zweimaliges Betätigen der Organisationstaste [↑] der Ast bis zur Betriebsart „Aufnahme" in umgekehrter Richtung durchschritten. Danach hat der Bediener die Organisationstaste [→] zu betätigen und ruft damit die gewünschte Betriebsart auf. Wird in der Folge die Taste [↓] gedrückt und erfolgt die Quittierung, gelangt die Steuerung in den Zustand der Wiedergabebereitschaft („Hydraulikwegeventil ein").

Alle Betriebsarten lassen sich in der für das Löschen eines Programms und der Herstellung der Wiedergabebereitschaft beschriebenen Art und Weise aufrufen.

Die automatisch ablaufende und zu einer bestimmten Betriebsart (z.B. Wiedergabebereitschaft) führenden Bedienung ist möglich, wenn dies im Rahmen der Anwenderinitialisierung vorgenommen und als „Präambel" auf Kassettenmagnetband gespeichert wurde. Dies ist in allen den Fällen vorteilhaft anwendbar, bei denen nach Einschalten des Roboters eine Programmwiedergabe mit einer bestimmten Programmnummer erfolgt.

Befehlssatz der Steuereinrichtung ursalog 5021

In der Betriebsart „Aufnahme" werden Informationen über die Bewegung/Geometrie und über das Ablaufverhalten/Aktionsverhalten in den Anwenderspeicher der Steuerung eingeschrieben. In diesem Speicher können bis zu 128 unterschiedliche Programme variabler Länge abgelegt werden. Die Gesamtdauer der Programme wird durch die zur Verfügung stehende Speicherkapazität und den gewählten Taktteiler für den Interpolator bestimmt und kann aus Tafel 8.7 entnommen werden. Jedes Programm besteht aus einer endlichen Anzahl von Anweisungen, deren prinzipieller Aufbau im Bild 8.12 angegeben ist. Der

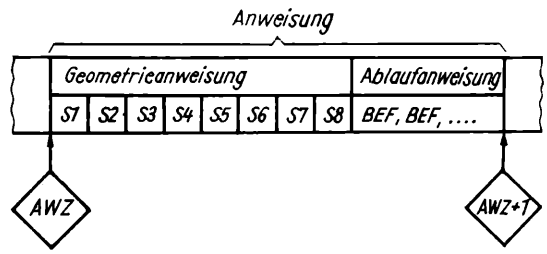
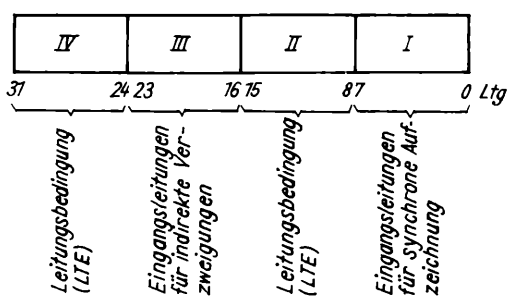


Bild 8.12
Prinzipieller Aufbau einer Anweisung
S1 bis *S8* Positionsstützwerte; *BEF* Befehl; *AWZ* Anweisungszählerstand



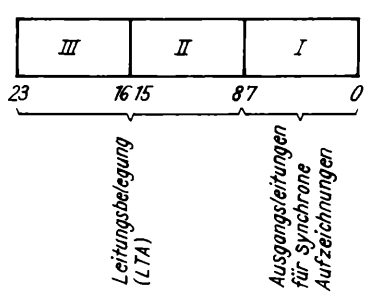


Bild 8.13
Beispiel einer Zuordnung von binären Ein- und Ausgängen zu den Operanden der Verzweigungsbefehle

Anweisungszähler, ein Zeiger, der die aktuelle Anweisung adressiert, sorgt für das Einschreiben (Auslesen bei der Wiedergabe) in einer steigenden Folge von virtuellen Adressen. Eine Einzelanweisung setzt sich aus der Geometrieanweisung und der (möglichen)

Tafel 8.3. Befehlssatz der Steuereinrichtung ursalog 5021

Befehlsgruppe	Mnemonik Operations-Operandkode	Wirkung des Befehls
Synchronisation des Anweisungszählers	HU Halt unbedingt	Der Anweisungszähler wird angehalten. Ein erneuter Programmstart ist nur durch eine Bedienerhandlung möglich.
	HB LTE Halt bedingt	Der Anweisungszähler wird bedingt angehalten. Die Bedingung ist für die ausgewählten Leitungen im Operanden anzugeben. Ein erneuter Programmstart ist nur durch eine Bedienerhandlung möglich.
	LB LTE Lauf bedingt	Der Anweisungszähler wird bedingt gestartet. Die Bedingung ist für die ausgewählten Leitungen im Operanden anzugeben.
	WB LTE Warten bedingt	Der Anweisungszähler wird so lange angehalten, bis die im Operanden anzugebende Leistungsbedingung erfüllt ist.
Veränderung des Anweisungszählers (Verzweigungen)	SUD PNR Sprung unbedingt direkt	Der Anweisungszähler wird auf die im Operanden anzugebende Programmnummer gestellt.
	SBD LTE, PNR Sprung bedingt direkt	Der Anweisungszähler wird auf die im 2. Operanden anzugebende Programmnummer gestellt, wenn die im 1. Operanden anzugebende Leitungsbedingung erfüllt ist.
	SUI Sprung unbedingt indirekt	Der Anweisungszähler wird auf die Programmnummer gestellt, die an ausgewählten Leitungen vorzugeben ist. Die Festlegung der dafür vorgesehenen Eingangsleitungen muß im Rahmen der Grundinitialisierung der Steuerung erfolgen.
	SBI LTE Sprung bedingt indirekt	Der Anweisungszähler wird auf die Programmnummer gestellt, die an ausgewählten Leitungen (siehe SUI) vorzugeben ist, wenn die im Operanden anzugebende Leitungsbedingung erfüllt ist.
	RUD PNR Ruf unbedingt direkt	Der Anweisungszähler wird auf die im Operanden anzugebende Unterprogrammnummer gestellt. Der Anweisungszählerstand +1 wird im Stapelspeicher abgelegt.
	RBD LTE, PNR Ruf bedingt direkt	Der Anweisungszähler wird auf die im 2. Operanden anzugebende Unterprogrammnummer gestellt, wenn die im 1. Operanden anzugebende Leitungsbedingung erfüllt ist. Der Anweisungszählerstand +1 wird im Stapelspeicher abgelegt.
	RUI Ruf unbedingt indirekt	Der Anweisungszähler wird auf die Unterprogrammnummer gestellt, die an ausgewählten Leitungen vorzugeben ist. Die Festlegung der dafür vorgesehenen Eingangsleitungen muß im Rahmen der Grundinitialisierung der Steuerung erfolgen. Der Anweisungszählerstand +1 wird im Stapelspeicher abgelegt.

Tafel 8.3 (Fortsetzung)

Befehlsgruppe	Mnemonik Operations-Operandkode	Wirkung des Befehls
	RBI LTE **Ruf bedingt indirekt**	Der Anweisungszähler wird auf die Unterprogrammnummer gestellt, die an ausgewählten Leitungen (siehe RUI) vorzugeben ist, wenn die im Operanden anzugebende Leitungsbedingung erfüllt ist. Der Anweisungszählerstand +1 wird im Stapelspeicher abgelegt.
	RU **Rückkehr unbedingt**	Der Anweisungszähler wird auf den zuletzt im Stapelspeicher abgelegten Anweisungszählerstand gestellt.
	RB LTE **Rückkehr bedingt**	Der Anweisungszähler wird auf den zuletzt im Stapelspeicher abgelegten Anweisungszählerstand gestellt, wenn die im Operanden anzugebende Leitungsbedingung erfüllt ist.
Verzögerung des Anweisungszählers	VZ ZEIT **Verzögerung, Zeit**	Der Anweisungszähler wird für die im Operanden anzugebende Zeit angehalten. Nach Ablauf der Zeit wird der Anweisungszähler gestartet.
	VT TAKT **Verzögerung, Takt**	Der Anweisungszähler wird für die im Operanden anzugebende Anzahl von externen Takten angehalten. Danach wird der Anweisungszähler gestartet.
Prozeß- Ein-/Ausgabe	AU LTA **Ausgabe unbedingt**	Die für eine Ausgabe vorgesehenen Leitungen einschließlich deren binärer Belegung ist im Operanden vorzugeben. Die Ausgabe bleibt bis zum nächsten Ausgabebefehl erhalten.
	ADZ ZEIT, LTA **Ausgabe, dynamisch Zeit**	Die Ausgabe der im 2. Operanden vorzusehenden binären Leitungsbelegung erfolgt für die Dauer der im 1. Operanden angegebenen Zeit.
	ADT TAKT, LTA **Ausgabe, dynamisch Takt**	Die Ausgabe der im 2. Operanden vorzusehenden binären Leitungsbelegung erfolgt für die Dauer der im 1. Operanden anzugebenden Anzahl von externen Takten (prozeßsynchrone Ausgabe).
	ABE LTE, LTA **Ausgabe, bedingt Eingang**	Die Ausgabe der im 2. Operanden vorzusehenden binären Leitungsbelegung erfolgt, wenn die im 1. Operanden anzugebende Leitungsbedingung erfüllt ist.
	ANW SW, i **Anweisung**; $i \in \{1 \ldots 8\}$ BEF, BEF	Dieser Befehl bewirkt die Ausführung einer Geometrieanweisung einschließlich möglicher Ablaufanweisung(en). Die Stützwerte SW sind durch Bedienhandlungen einzugeben. Es können alle Ablaufanweisungen entsprechend dem Befehlssatz Anwendung finden.

Ablaufanweisung zusammen. Die Geometrieanweisungen werden durch Vorführen der Bewegungsbahn (direktes Teach-in) erzeugt und repräsentieren je einen Stützwert S_i für jede Achse des kinematischen Systems (Raumpunkt). Der zeitliche Abstand zwischen den

Stützwerten ist von dem gewählten Taktteiler und der internen bzw. externen Taktfrequenz abhängig. Die Ablaufanweisungen können sowohl programmorganisatorische als auch das Aktionsverhalten bestimmende Befehle enthalten. Diese werden von dem Programmierer in den meisten Fällen erst nach der Aufnahme der Bewegungsbahn eingegeben. Eine Ausnahme bilden die Ablaufanweisungen, bei denen die Belegung von binären Eingängen prozeßsynchron aufgezeichnet werden sollen. Die Generierung dieser Befehle erfolgt direkt bei der Aufnahme der Bewegungsbahn. Die Menge aller für den Anwender frei verfügbaren Befehle wird als Befehlssatz der Steuereinrichtung bezeichnet und ist in Tafel 8.3 angegeben. Die linke Spalte in der Tabelle definiert die Befehlsgruppe; die Spalte Mnemonik gibt die für die Befehlseingabe vereinbarten Operationskodes und Operanden an. In der rechten Spalte ist die Wirkung des Befehls beschrieben.

Die bedingten Befehle beziehen sich auf die an den binären Eingängen der Steuereinrichtung anliegenden Informationen. Im Bild 8.13 wird das allgemeine Anschlußschema sowohl der binären Eingangsleitungen (LTE) als auch das der binären Ausgangsleitungen (LTA) gezeigt. Im Rahmen der Anwenderinitialisierung der Steuerung können bestimmte Ein- und Ausgänge roboterspezifisch von einer Befehlsbeeinflussung ausgeschlossen werden. Bei der Verwendung von indirekten Verzweigungsbefehlen ist ebenfalls durch die Anwenderinitialisierung ein 8-Bit-Eingangsblock für die Bereitstellung der (Unter-)Programmnummer festzulegen.

Die Eingabe der Operanden erfolgt über die Tastatur entsprechend der Bedienerführung. Bei Anwendung des Operanden LTE ist zu beachten, daß die Notierung der Verknüpfung in einer disjunktiven Normalform erfolgen muß (s. auch Bild 8.44b). Soll z.B. die Programmabarbeitung so lange gestoppt werden, bis auf der Eingangsleitung E30 1-Signal oder auf den Eingangsleitungen E9 0-Signal und auf E8 1-Signal anliegt, so ist für den Operanden des Befehls WB zu notieren und einzugeben: E30/E9E8. Das Operationszeichen für die Disjunktion ist ein Leerschritt zwischen den Implikanten, während für die Negation ein Schrägstrich vor der betreffenden Leitungsnummer einzugeben ist.

8.4.3. Funktionsbeschreibung

Aufbauend auf die prinzipielle Struktur einer Industrierobotersteuerung, wie sie im Abschnitt 2. behandelt ist, ergeben sich folgende relativ autonome Funktionsblöcke:

- Funktionseinheit zur Verwaltung des Speichers für das Anwenderprogramm und zur Befehlsinterpretation,
- Funktionseinheit zur Kommunikation mit dem Bediener/Programmierer,
- Funktionseinheit zur Glättungsinterpolation und Servoregelung (Lageregelung).

Sowohl zur Realisierung einer modularen Steuerungskonzeption als auch für eine zweckmäßige Aufgabenverteilung wird bei der Steuerung ursalog 5021 jedem dieser Funktionsblöcke ein eigener Mikrorechner zugeordnet und diese in einer Master-Slave-Struktur nach Bild 8.14 zusammengefaßt. Die gewählte Torkopplung nutzt die auf der Rechnerkarte K2521 vorhandene Zusatzlogik [8.1], zwingt aber aufgrund der begrenzten Übertragungsrate von etwa 30 KByte/s zu einem weitgehend autonomen Arbeiten der Einzelrechner.

Die Organisation der Kopplung erfolgt durch spezielle Koppelprogramme, die sowohl im Master als auch in den Slaves resident sind. Dabei ist dem Master die höchste Priorität und dem Slave 2 die niedrigste Priorität zugeordnet. Es können unter Beachtung der Prioritäten Koppelanforderungen vom Master an einen der Slaves sowie von einem Slave

an den Master bearbeitet werden. Eine direkte Anforderung von Slave 1 an Slave 2 ist nicht möglich und muß über einen Masterruf abgewickelt werden. Die Koppelprogramme greifen im wesentlichen auf die sich in jedem Einzelrechner befindlichen und den aktuellen Zustand der Steuerung widerspiegelnden Abbildspeicher zu. Prinzipiell werden vier Arten von Kopplungen unterschieden.

1. Datentransfer – Schreiben

Aus dem Abbildspeicher des Senders werden Informationen in die äquivalenten Bereiche des Abbildspeichers des Empfängers übertragen. Der Empfänger reagiert nicht unmittelbar auf die übertragene Information, sondern erst, wenn er zufällig darauf zugreift oder durch ein Kommando dazu aufgefordert wird.

2. Datentransfer – Lesen

Der Sender ruft Informationen aus dem Abbildspeicher des Empfängers ab und trägt sie in die äquivalenten Bereiche seines eigenen Abbildspeichers ein. Der Empfänger erhält selbst keine Kenntnis über den erfolgten Speicherzugriff.

3. Kommando ohne Datentransfer

Der Sender löst eine Kommandoroutine beim Empfänger aus, die unter möglicher Nutzung von Kommandoparametern ggf. auch wieder auf den Sender zurückwirken kann.

4. Kommando mit Datentransfer

Der Sender löst eine Kommandoroutine beim Empfänger aus und überträgt dazu Daten. Auf der Empfängerseite werden die Daten im Block übernommen und ausgewertet.

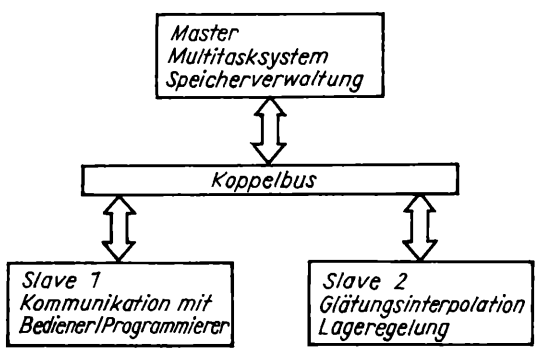


Bild 8.14
Master-Slave-Struktur der Industrieroboter-steuerung ursalog 5021

Funktionsumfang Master

Die Hardwarekomponenten des Master- oder Leitrechners sind im Bild 8.15 angegeben und arbeiten auf den standardisierten Systembus des Mikrorechnersystems K1520. Sie bestehen im einzelnen aus der Rechnerkarte ZRE K2521 mit dem Prozessor U880 und der kombinierten RAM/ROM-Speicherkarte OFS K3620. Auf diesen beiden Karten sind die Funktionalprogramme des Masters gespeichert. Für die Speicherung des Roboterprogramms stehen bis zu vier TEST RAM (dynamische Speicher) mit je 64 KByte zur Verfügung. Die Adressierung (Auswahl) dieser Speicherkarten einschließlich ihrer Segmentierung erfolgt über E/A-Adressen. Über die Anschlußsteuerung AKB K5020 wird das für die Datenarchivierung vorgesehene Kassettenmagnetbandgerät K5200 betrieben. Zwei Steckeinheiten DFS 12 dienen zur Eingabe binärer Informationen über Optokopp-

ler (maximal 32 Kanäle). Die Steckeinheit DA-R gestattet die potentialfreie Ausgabe binärer Informationen (maximal 24 Kanäle), während die Anschlußsteuerung IFSS (*I*nter*f*ace *s*ternförmig *s*eriell) zur Kopplung mit einem übergeordneten Rechner dient.

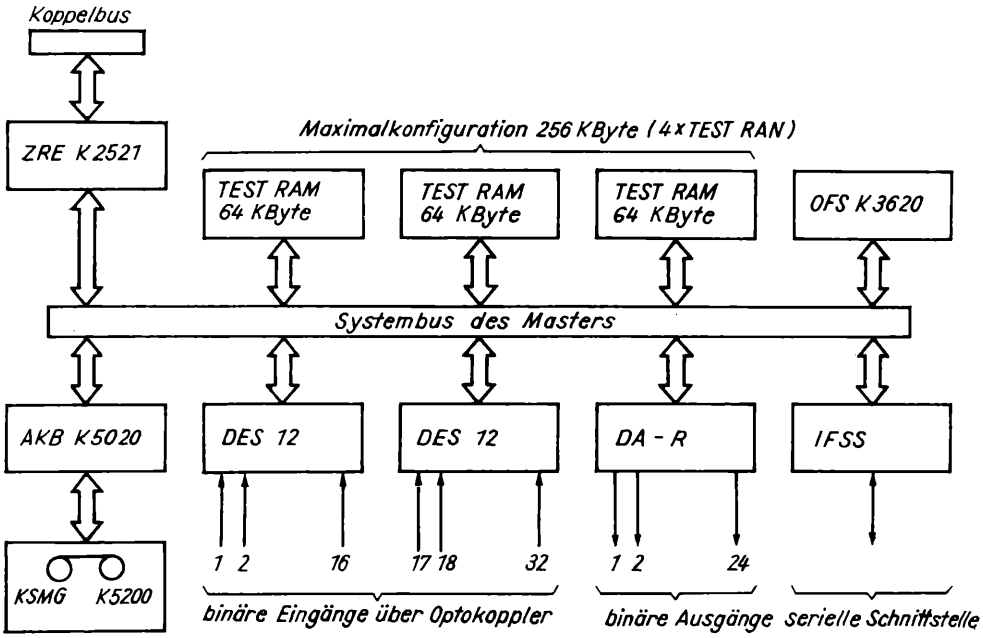


Bild 8.15. Hardwarekomponenten des Masters

Im Masterrechner sind im wesentlichen folgende Programme implementiert:

- Echtzeitbetriebssystem der Steuerung,
- Programme zur Verwaltung des Wegedaten- und Befehlsspeichers,
- Programme zur Interpretation und Ausführung der Anwenderbefehle,
- Programme zur Archivierung des Roboterprogramms auf Kassettenmagnetbandgerät.

Das *Echtzeitbetriebssystem* stellt ein neutrales, von der eigentlich zu realisierenden Aufgabe völlig unabhängiges Programmpaket dar. Es koordiniert alle Rechenprozesse, die den Funktionsinhalt der Steuerung realisieren. Sowohl die im Master als auch die in den Slaves residenten Programme stellen für das Echtzeitbetriebssystem Tasks dar, die seiner Verwaltung und Koordinierung unterliegen.

Die Programme zur *Verwaltung des Wegedaten- und Befehlsspeichers* ordnen den physikalischen Speichern (in der maximalen Ausbaustufe 256 KByte) den virtuellen Roboterprogrammspeicher zu und organisieren das Beschreiben und das Auslesen von Programmen sowie das Einfügen und das Entfernen von Einzelanweisungen. In der ersten Phase erfolgt im Rahmen der Systeminitialisierung ein Speichertest, der evtl. fehlerhafte Speicher erkennt und diese in Segmenten zu 4 KByte von einer möglichen Programmaufzeichnung ausschließt. Alle funktionsfähigen Speichersegmente werden danach zu einem Speicherblock gebunden, der dem Anwender sowohl zur Programmaufzeichnung als auch zur Übernahme von bereits erstellten Programmen aus dem Kassettenmagnetbandgerät zur Verfügung steht. Wie im Bild 8.16b dargestellt, wird das bei einer Programmaufnahme neuerstellte Programm an bereits vorhandene angefügt. Bei der Wiedergabe kann ein beliebiger, ein vollständiges Programm repräsentierender Abschnitt ausgelesen werden. Im

Bild 8.16c wird das Einfügen und das Entfernen von Einzelanweisungen bzw. Befehlen, das i. allg. nach jeder Programmaufnahme erfolgt, gezeigt. Die Struktur des Speichers für das Roboterprogramm wird im Bild 8.17 angegeben. Jedes in sich abgeschlossene Programm beginnt mit der virtuellen Adresse Null. Zu seiner Kennzeichnung wird eine Marke, die mit dem Programmnamen identisch ist und unter der es im Speicherraum zu

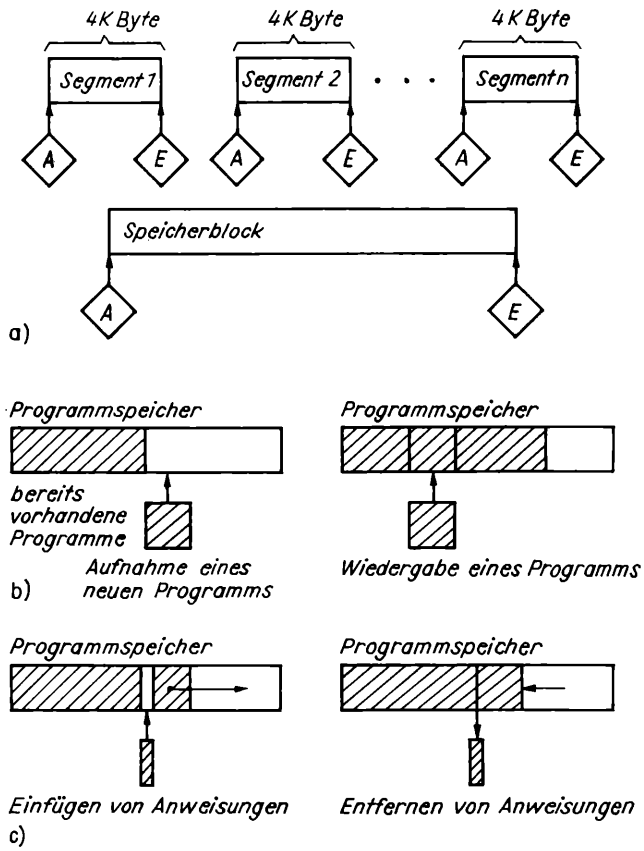


Bild 8.16
Organisation des Anwenderspeichers

a) Zusammenfügen des Speicherblocks aus Segmenten zu je 4 kbyte; b) Speicherorganisation bei Aufnahme und bei Wiedergabe eines Programms; c) Einfügen und Entfernen von Anweisungen aus dem Speicher

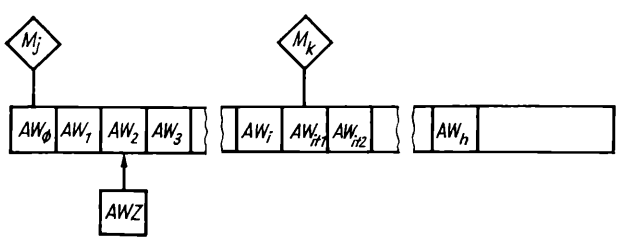


Bild 8.17
Prinzipielle Struktur des Anwenderspeichers

M_j, M_k Programmarken/Programmnamen; AW Anweisungen; AWZ Anweisungszählerstand

finden ist, festgelegt (symbolische Adressierung). Bei der Programmaufnahme werden Anweisungen mit steigender Folge im Speicher abgelegt; ihre Adressierung geschieht mit Hilfe des Anweisungszählers. Bei der Wiedergabe erfolgt die Abarbeitung der Anweisungen in der gespeicherten Reihenfolge; ein Verlassen dieser Folge ist nur bei Ausführung eines Verzweigungsbefehls möglich (s. Tafel 8.8).

Das Format einer Anweisung wird im Bild 8.18 gezeigt. Eine Anweisung besteht generell aus einer Geometrie- und einer möglichen Ablaufanweisung. Die Geometrieanwei-

sung enthält die zu einem bestimmten Anweisungszählerstand zugeordneten Wegedaten für alle Achsen, während die Ablaufanweisung aus Befehlen entsprechend dem Befehlssatz der Steuerung besteht. Bis auf die prozeßsynchrone Aufzeichnung von binären Eingangssignalen erfolgt die Programmierung von Ablaufanweisungen im Sinne des Einfügens von Anweisungen/Befehlen. Zur Vereinfachung der internen Verarbeitung von

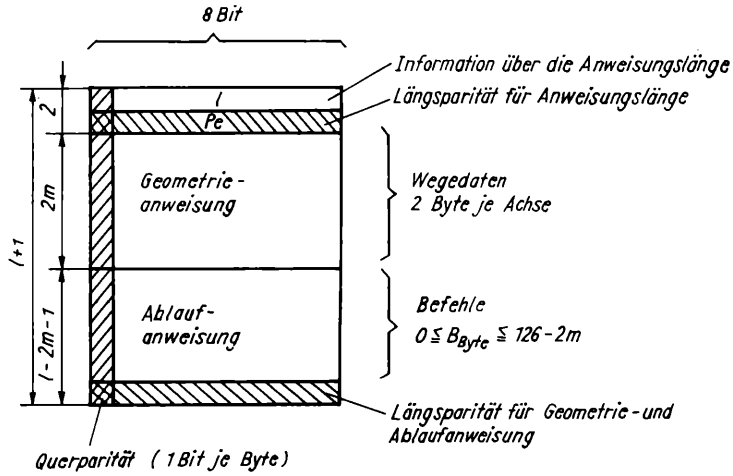


Bild 8.18
Aufbau einer Anweisung
l Anweisungszählerlänge; m Anzahl der Achsen; B_{Byte} Anzahl der für die Speicherung von Befehlen zur Verfügung stehenden Speicherbyte in einer Anweisung

Anweisungen, vor allem bei der Suche nach Marken in einem abgeschlossenen Programm, ist jeder Anweisung eine Information über ihre Länge vorangestellt. Um die Zuverlässigkeit des Gesamtsystems zu erhöhen, werden bei der Speicherung der Anweisungen eine Reihe von zusätzlichen Informationen zur Datensicherung abgelegt. Generell wird für jedes Byte ein Paritätsbit erzeugt und gespeichert (Querparität). Außerdem werden der Information über die Länge der Anweisung und dem Block, bestehend aus Geometrie- und Ablaufanweisung, je ein Byte angefügt (Längsparität). Mit dieser Maßnahme läßt sich bei der Wiedergabe ein auftretender „Einbitfehler" in der Information über die Länge der Anweisung und innerhalb des Blockes Geometrie- und Ablaufanweisung automatisch korrigieren. Fehler, die durch „Mehrbitfehler" bedingt sind, werden erkannt und je nach Lokalisation eine entsprechende Reaktion ausgelöst.

Die Programme zur *Interpretation und Ausführung der Anwenderbefehle* (Ablaufanweisungen) realisieren den Befehlssatz der Steuerung und beziehen sich hauptsächlich auf die Verarbeitung von binären Eingangssignalen und der Ausgabe von Binärinformationen an den Prozeß über die Hardwarekomponenten DES 12 und DA-R. In Tafel 8.4 ist die mnemotechnische Darstellung der Befehle und Operanden und die Kodierung der Befehle in sedezimaler Schreibweise angegeben. Die interne Kodierung der Operanden LTE, LTA, PNR, ZEIT und TAKT ist in Tafel 8.5 dargestellt und dient in dieser Form zur programmtechnischen Realisierung der Anwenderbefehle.

Im ersten Byte des Operanden LTE ist die Anzahl der in einer disjunktiven Normalform konjunktiv und disjunktiv zu verknüpfenden Eingangsleitungen vorangestellt. Mit einem Befehl lassen sich bis zu 128 Eingangsbelegungen verknüpfen. Im Beispiel ist dies für sechs Verknüpfungen gezeigt. Im Bit 5 des Operanden ist der Operator für die Negation und im Bit 6 der Operator für die nächstfolgende Operation abgelegt. Sowohl in diesem als auch bei allen anderen Operanden gibt eine 1 im Bit 6 des letzten Byte das Ende des Operanden an. In Tafel 8.5 wird der Operand LTA für eine binäre Ausgabe gezeigt. In den Bits ∅ bis 4 ist jeweils die binäre Kodierung des auszugebenden Kanals (Ausgangsleitung) abgelegt, während Bit 5 festlegt, ob der Kanal ein- oder ausgeschaltet werden soll.

Tafel 8.4. Sedezimale Kodierung der Befehle der Steuereinrichtung ursalog 5021

low ↓ / high →	Ø	1	2	3
Ø	HU	AU (LTA)		
1	HB (LTE)	ADZ (ZEIT, LTA)		
2	LB (LTE)	ADT (TAKT, LTA)		
3	WB (LTE)	ABE (LTE, LTA)		
4	SUD (PNR)			
5	SBD (LTE, PNR)			
6	SUI			
7	SBI (LTE)	ANW		
8	RUD (PNR)			
9	RBD (LTE, PNR)			
A	RUI			
B	RBI (LTE)	Raum für Erweiterungen		
C	RU			
D	RB (LTE)			
E	VZ (ZEIT)			
F	VT (TAKT)			

Bedeutung der Abkürzungen (außer ANW Anweisung):

1. Buchstabe	2. Buchstabe	3. Buchstabe
H Halt	U unbedingt	D direkt
L Lauf	B bedingt	I indirekt
S Sprung	Z Zeit	E Eingang
R Ruf/Rückkehr	T Takt	
V Verzögerung		
A Ausgabe		

Tafel 8.5. Interne Kodierung der Operanden LTE, PNR, ZEIT und TAKT

Operand	Allgemeiner Aufbau	Kodierungsbeispiel
LTE	6 5 4 3 2 1 k 1 D_1 C_1 E_1 2 D_2 C_2 E_2 k D_k C_k E_k Paritätsbit K Anzahl der kunjunktiv und disjunktiv zu verknüpfenden Eingangsleitungsbelegungen $1 < k \leq 127$; k = NZ E Binäre Kodierung der Nummer der Eingangsleitung C Operator: 1 → Negation D Operator: 0 → konjunktion 1 → Disjunktion	LTE $\underbrace{E30}_{1.}$ v. $\underbrace{E24 / E15\ E14}_{2.}$ v $\underbrace{/E9\ E8}_{3.}$ Implikant 0 0 0 0 1 1 0 } 1. Implikant 1 0 1 1 1 1 0 0 0 1 1 0 0 0 0 1 0 1 1 1 1 } 2. Implikant 1 0 0 1 1 1 0 0 1 0 1 0 0 1 } 3. Implikant 1 0 0 1 0 0 1
LTA	6 5 4 3 2 1 0 1 0 B_1 A_1 2 0 B_2 A_2 3 0 B_3 A_3 k 1 B_k A_k Ende des Operanden A Binäre Kodierung der Nummer der Ausgangsleitung B Belegung der Ausgangsleitung 0 → Ausschalten 1 → Einschalten	Leitungen A_2, A_4, A_{10} Einschalten Leitung A_{14} Ausschalten 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 1 0 1 0 1 0 1 0 0 1 1 1 0
PNR	6 5 4 3 2 1 0 PNR PNR Symbolische Adresse (Marke) für Programm $0 \leq PNK \leq 127$ PNK = NZ	Programmnummer 54 0 1 1 0 1 1 0
ZEIT	6 5 4 3 2 1 0 $ZEIT_0$ $ZEIT_1$ 13 12 11 10 9 8 7 ZEIT Faktor für die Zeitbasis von 10ms	Zeitverzögerung von 30.00s 0 1 1 1 0 0 0 0 0 1 0 1 1 1
Takt	6 5 4 3 2 1 0 $ZEIT_0$ $ZEIT_1$ 13 12 11 10 9 8 7 TAKT Anzahl der zur Verzögerung dienenden Takte $1 \leq TAKT \leq 16383$	Verzögerung um 332 Takte 1 0 0 1 1 0 0 0 0 0 0 0 1 0

Der 1-Byte-Operand PNR läßt die Auswahl von maximal 128 Einzelprogrammen bzw. Marken in einem Anwenderprogramm zu. Außerdem sind in Tafel 8.5 die Operanden für die Befehle angegeben, die ein zeitweiliges Stillsetzen des Anweisungszählers bewirken.

Das Programmpaket zur *Archivierung des Roboterprogramms* auf Kassettenmagnetbandgerät ermöglicht den blockweisen Datentransfer zwischen dem Anwenderspeicher und dem Bandgerät K5200 [8.25]. Zur Vermeidung von Übertragungsfehlern erfolgt ein Kontrollesen der aufgenommenen Daten beim Aufzeichnen (read after write). Zusätzlich wird bei der Aufzeichnung eines Blockes ein CRC-Zeichen (*c*ycling *r*edundancy *c*hecking) gebildet und im Anschluß an die auf Band abgespeicherten Daten als 2-byte-Wort abgelegt. Beim Lesen des Blockes vom Band wird erneut das CRC-Zeichen gebildet und mit dem gespeicherten verglichen. Bei Übereinstimmung werden die Daten als gültig erkannt und in den Anwenderspeicher übergeben. Im anderen Fall erfolgt eine Fehlerkorrektur durch nochmaliges Lesen der Daten vom Band. Ist nach einer bestimmten Anzahl von Wiederholungen keine Übereinstimmung der CRC-Zeichen zu erreichen, wird eine Fehlermeldung ausgelöst.

Funktionsumfang des Slaves 1

Der Slave 1 dient als Bedienrechner zur Kommunikation mit dem Bediener/Programmierer; seine Hardwarekomponenten sind im Bild 8.19 dargestellt. Mit Hilfe einer alphanumerischen Tastatur werden alle Bedienhandlungen (außer Not-Aus) durchgeführt,

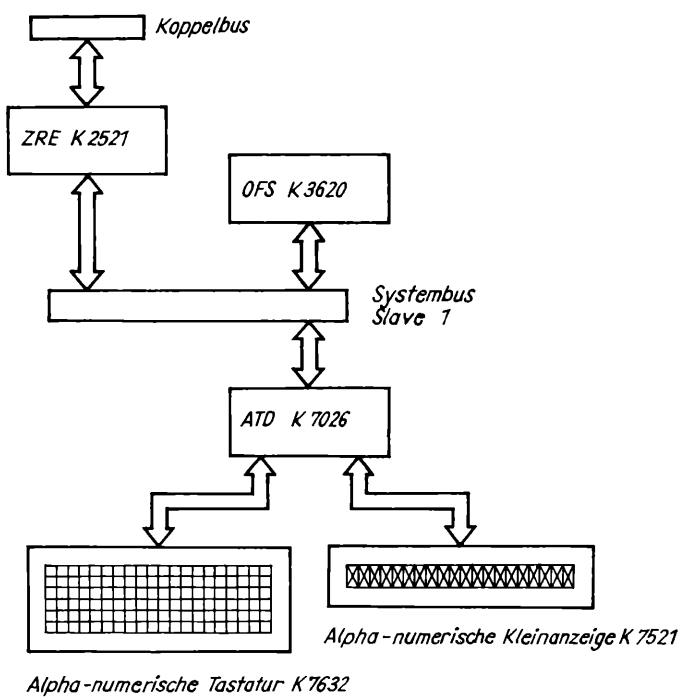


Bild 8.19
Hardwarestruktur von Slave 1

während die Ausgabe von Kurztexten, die Anzeige von eingegebenen Kommandos und typischen Zuständen der Steuereinrichtung (z. B. Fehlermeldungen) über eine alphanumerische Anzeigezeile mit 5 × 7-Punktmatrix-Anzeigeelementen erfolgt. Die Anschlußsteuerung K7632 dient zur Ansteuerung der Tastatur K7632 und der 32stelligen

Kleinanzeige ANK 7521, während die Kartenbaugruppe OFS K3620 zur Speicherung der benötigten Verwaltungsprogramme benötigt wird.

Die Softwarekomponenten realisieren die im Rahmen der Bedienerführung vorgesehene Menütechnik mit der dafür notwendigen umfangreichen Zeichenkettenverarbeitung und Textanzeige. Mit Hilfe von Verzweigungstabellen werden die in einer Baumstruktur angeordneten Ebenen der einzelnen Aktionsprogramme erreicht.

Funktionsumfang des Slaves 2

In der Phase der Programmaufnahme dient Slave 2 zur Aufbereitung der von den Meßsystemen des kinematischen Systems gewonnenen Informationen und übergibt diese zur zeitäquidistanten Speicherung dem Master. In der Wiedergabephase erhält er die gespeicherten Werte der Bahnkurve (Stützwerte) und generiert zusätzliche Bahnpunkte. Stützwerte und zusätzliche Bahnpunkte fungieren als Sollwerte für die Servoregelung.

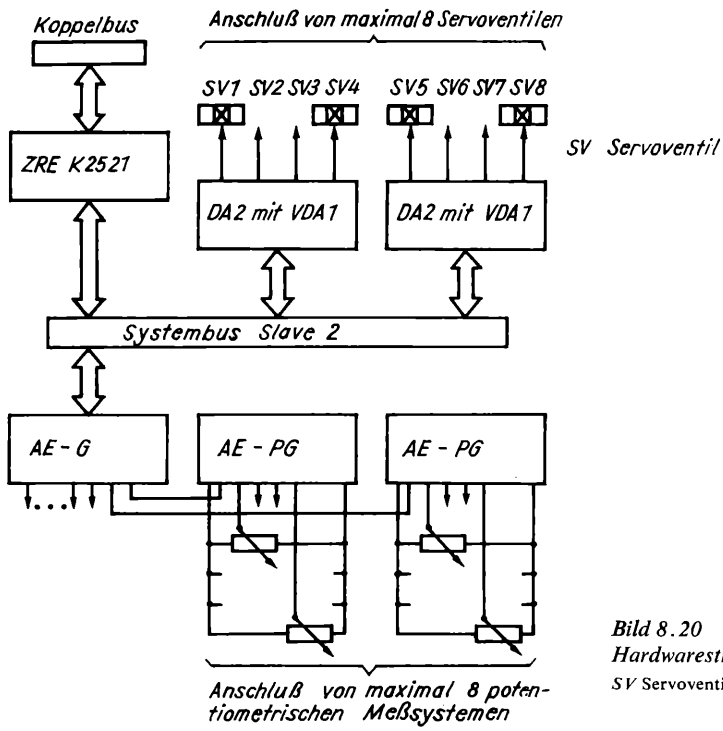


Bild 8.20
Hardwarestruktur von Slave 2
SV Servoventil

Die Hardwarestruktur des Slaves 2 ist im Bild 8.20 dargestellt. Die analoge Meßwerterfassung erfolgt mit Hilfe von zwei Anschlußeinheiten für passive Geber (AE-PG). In einer Modifizierung sorgen sie für die Bereitstellung einer hochstabilen Versorgungsspannung für die Meßpotentiometer (Plastschichtpotentiometer). Zur möglichst vollständigen Ausnutzung des Auflösebereichs des Analog-Digital-Wandlers bei eingeschränktem Betätigungswinkel der Potentiometer (direkter Einbau der Meßsysteme in die Gelenke des Roboters) erfolgt auf den obengenannten Anschlußeinheiten eine entsprechende Korrektur. Die schaltungstechnische Konzeption sowohl der Anschlußeinheiten für passive Geber als auch die der Grundkarte (AE-G) mit dem 8-Kanal-Analogmultiplexer, dem Instrumentationsverstärker und dem 12-bit-Analog-Digital-Wandler sorgen für eine hohe Gleichtaktunterdrückung. Die Kartenbaugruppen DA 2 dienen zur Bereit-

stellung der Spannung für die Geschwindigkeitssollwerte und in Verbindung mit den Vorsätzen VDA 2 zur Stromansteuerung der Servoventile. Die Informationsbreite der errechneten Geschwindigkeitssollwerte beträgt 8 Bit. Die programmtechnischen Komponenten des Slaves 2 bestehen vor allem aus der für die Servoregelung (Lageregelung) einschließlich Positionsmeßwerterfassung und Geschwindigkeitssollwertausgabe und der für die Interpolation.

Die Programme für die Servoregelung, die ein proportionales Verhalten der einzelnen Regler realisieren, sind nur in der Betriebsart „Wiedergabe" aktiviert und garantieren eine stetige Ausgabe von Geschwindigkeitssollwerten im Zeitraster von 10 ms. Die Verstärkungsfaktoren für die den einzelnen Achsen zugeordneten Regler werden im Rahmen der Grundinitialisierung einmalig in den Abbildspeicher der Steuerung eingeschrieben und können während des Betriebs je nach der Stellung des kinematischen Systems verändert werden (adaptive Regelung). Die Sollwerte für die einzelnen Regler werden durch den Interpolator oder durch direkte Verwendung der aktuellen Positionsistwerte bereitgestellt. Das verhindert ein unkontrolliertes Verhalten der Kinematik nach dem Einschalten der Betriebsart Wiedergabe und bei Anfahrstrategien. Zur Realisierung der Interpolation 3. Grades (Einfügen von Ästen einer Parabel 3. Grades zwischen den Stützstellen) ist die Kenntnis von jeweils vier Stützstellen notwendig. Diese werden durch den Master gemeinsam mit dem Kommando „Interpolation" an den Slave 2 übergeben und dort zwischengespeichert. Anschließend erfolgt die Generierung der durch den Taktteiler vorgegebenen Anzahl von Zwischenwerten im zentralen Intervall und deren Übergabe an den Lageregler. Die zeitliche Aufeinanderfolge der Stützwertbereitstellung aus dem Arbeitsspeicher (Roboterprogramm) des Masters erfolgt entweder intern oder extern taktgesteuert. Letztere Möglichkeit erlaubt die Zeitraffung bzw. die Zeitdehnung gegenüber dem Zeitbezug der aufgenommenen Bahn.

Anhang

A1. Numerisches Steuerungssystem CNC 600

Die CNC 600 ist ein numerisches Bahnsteuerungssystem, mit dem technologische Vorgänge und Bewegungen vorzugsweise an Dreh-, Bohr-, Fräsmaschinen und Bearbeitungszentren gesteuert werden können.

Es können maximal fünf NC-gesteuerte Achsen bewegt werden, wobei die Interpolationsarten 2-D-linear und zirkular, 3-D-linear und Helixinterpolation realisiert sind.

Die Meßauflösung in den Achsen beträgt 0,001 mm bzw. 0,001°. Als Meßsysteme können sowohl phasenzyklisch-induktive (Linearinductiosyn, Rundinductosyn, Resolver) als auch inkrementale (translatorisch, rotatorisch) verwendet werden, die in zahlreichen Varianten miteinander kombiniert werden können. Über D/A-Wandler-Ausgänge können die Gleichstromachsantriebe angesteuert werden. Die maximalen Verfahrensgeschwindigkeiten (Eilgang) liegen bei 20 $m \cdot min^{-1}$.

Zur Realisierung der Anpaß- und weiterer Funktionen besitzt die CNC 600 eine integrierte, auf Mikrorechnerbasis realisierte PMC. Nur die eigentliche Ein-/Ausgabe-Steuerung (PEAS) ist außerhalb des Steuerungsschranks (Logikschrank) im Leistungsteil der Maschine untergebracht.

Eine komfortable Handhabung der Steuerung in allen Modi wird auf der Basis eines leistungsfähigen Bedien- und Anzeigekomplexes, der durch Bildschirm und ein umfangreiches Programmsystem unterstützt wird, erreicht.

Für Maschinensteuerfunktionen ist auch eine Zweistationenbedienung möglich. In den Steuerschrank ist der Leser zur Lochstreifeneingabe eingebaut; eine Steckdose für den Stanzer befindet sich in der Außenwand des Schrankes. Über einen On-line-Anschluß kann die CNC 600 in ein übergeordnetes Steuerungssystem integriert werden. Steuerungsinbetriebnahme, Inbetriebnahme von Anwenderprogrammen und Steuerungsdiagnose erfolgen ohne zusätzliche Geräte mit den Mitteln der Steuerung.

Die Vielzahl der möglichen Steuerungsvarianten läßt sich den beiden großen Gruppen „CNC 600 Bohren/Fräsen" und „CNC 600 Drehen" zuordnen.

Für Anwenderprogramme stehen für Varianten des Typs Bohren/Fräsen maximal 32 KByte Werkstückprogrammspeicher (ungestützt), maximal 5 KByte Unterprogrammspeicher (gestützt) und maximal 16 KByte PMC-Programmspeicher (EPROM) zur Verfügung.

Bei der Variante Drehen liegen diese Werte bei maximal 8 KByte für Werkstückprogrammspeicher (gestützt), maximal 6 KByte Unterprogrammspeicher (gestützt) und maximal 16 KByte PMC-Programmspeicher (EPROM).

Das Steuerungssystem CNC 600 konfiguriert sich aus

- Logikschrank,
- Prozeß-Ein-/Ausgabe-Steuerung PEAS,
- Maschinensteuertafel,
- Verbindungsleitungen.

Bild A1.1 veranschaulicht die Teile des Gesamtsystems. Die Prozeß-Ein-/Ausgabe-Steuerung ist für den Einbau in das Leistungsteil bestimmt. Die Maschinensteuertafel kann extern und/oder intern (d. h. am Logikschrank) angeordnet werden. Während der Lochstreifenleser fest im Logikschrank eingebaut ist, befindet sich in der Außenhaut des Logikschranks eine Steckdose für den Anschluß eines Lochstreifenstanzers (Kabellänge 3 m).

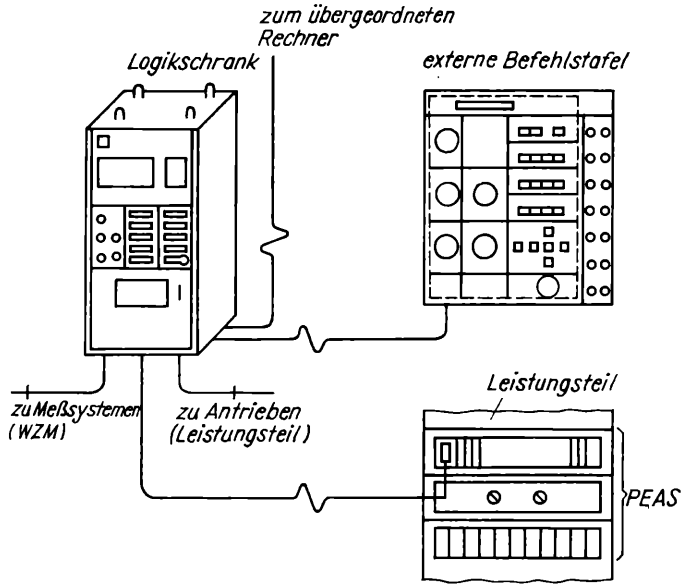


Bild A1.1
Gesamtsystem CNC 600

Der *Logikschrank CNC 600* enthält

- die Elektronik sowie die Bedienelemente (Taster, Drehwahlschalter) und Anzeige (Bildschirm) zur Korrespondenz mit dem Bediener; die Bedienblende des Steuerungsschranks ist dabei unterteilt in eine Datentafel und in eine wahlweise am CNC-Schrank angebrachte Maschinensteuertafel (Variante Bohren/Fräsen);
- die Elektronik zur Korrespondenz mit der Datenverarbeitungsperipherie (eingebauter Lochstreifenleser, anschließbarer Stanzer) und einem übergeordneten Rechner in On-line-Anschluß;
- die Elektronik zur Bedienung der prozeßseitigen Peripherie (PEAS, Meßsysteme, Antriebe);
- die gesamte Elektronik zur internen Datenverarbeitung, wobei die Anpaßsteuerung als programmierbare Steuerung (PMC) in den Steuerschrank integriert ist; die PMC korrespondiert über einen bidirektionalen Datenkanal mit der PEAS;
- Stromversorgung für die gesamte Logik sowie die Meßsysteme.

Die *Prozeß-Ein-/Ausgabe-Steuerung* (PEAS) realisiert die Ein-/Ausgabe von digitalen Signalen vom/zum Prozeß sowie die Ausgabe einer Analogspannung für die Steuerung des Hauptantriebs der Werkzeugmaschine. Sie besitzt eine eigene Stromversorgung sowie eine Lüfterbaugruppe und ist konstruktiv als Einbauvariante gestaltet.
Die *externe Maschinensteuertafel* kann auch vorhanden sein, wenn am Logikschrank bereits eine solche angebracht ist. Sie ist konstruktiv als Einbauvariante ausgeführt.

Die die Komponenten des Systems verbindenden Leitungen einschließlich derjenigen zu Meßsystemen und Antrieben haben eine maximale Länge von 50 m. Die Verbindungsleitung zum übergeordneten Rechner (On-line-Anschluß) kann eine Länge bis zu 500 m besitzen.

A1.1. Gerätetechnik des Logikschranks

Das im Bild A1.2 dargestellte Blockschaltbild vermittelt einen Überblick über die im Logikschrank der CNC 600 realisierten Funktionsmoduln, aus dem auch grob die Funktion, die nachfolgend genauer erläutert wird, zu erkennen ist. Das gesamte System stellt eine Multimikrorechneranordnung aus maximal neun Mikrorechnern, die alle auf der Basis des Mikrorechnersystems U880 aufgebaut sind, dar.

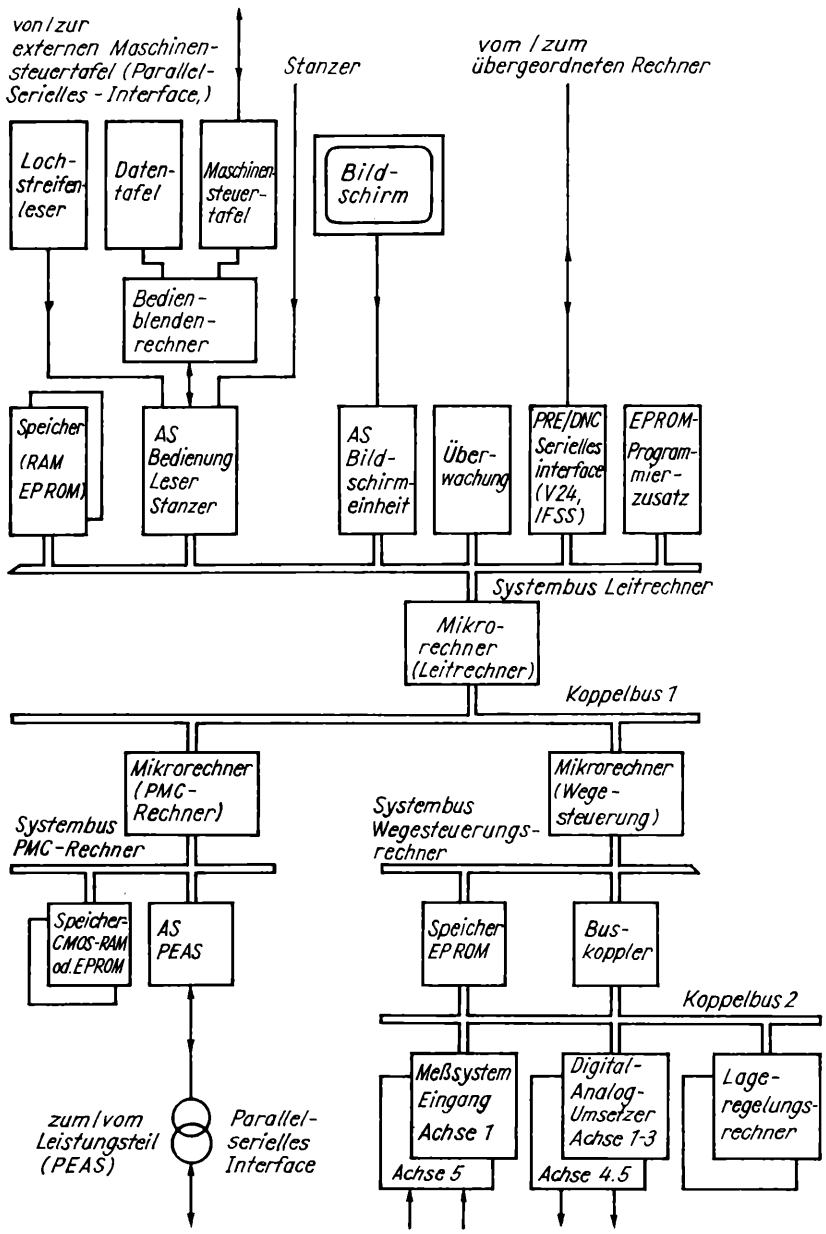


Bild A1.2. Blockschaltbild der CNC 600 (Logikschrank)

Die Gesamtanordnung läßt sich in einzelne Teilkomplexe unterteilen, die jeweils einzelnen Mikrorechnern bzw. Mikrorechnergruppen zuzuordnen sind:

- Komplex Leitrechner,
- Komplex Bedienblendenrechner,
- Komplex Wegesteuerungs- und Lageregelungsrechner,
- Komplex PMC-Rechner.

Das Zusammenwirken des Rechners wird durch die Rechnerkopplungen realisiert. Die Funktion der Steuerung in ihren verschiedenen Betriebsarten wird durch die in den einzelnen Rechnern fest installierten (residenten) bzw. vorübergehend geladenen (flüchtigen) Programmsysteme bestimmt.

A1.2. Komplex Leitrechner

Im Rahmen des Steuerungssystems hat der Komplex Leitrechner folgende Funktionen zu erfüllen:

Kommunikationsfunktionen

- Kommunikation mit dem Bediener und der DV-Peripherie,
- Kommunikation mit den angeschlossenen Slaverechnern für
- Wegesteuerung und PMC;

Informationsverarbeitungs- und Steuerungsfunktionen

- Ablaufsteuerung (Realisierung der Anpaßbedingungen),
- Satzaufbereitung (Realisierung der Programmieranleitung),
- Organisation der Bedienung und Anzeige (Realisierung der Bedienanleitung);

Speicherfunktionen

- Speicherung von Werkstückprogrammen,
- Speicherung von technologischen und werkstückspezifischen Unterprogrammen,
- Speicherung von Korrekturdaten,
- Speicherung von Generierdaten;

Systemüberwachungsfunktionen.

Kern der Hardware dieses Komplexes ist die zentrale Recheneinheit K2521 [8.1]. An dem von ihr gebildeten standardisierten Systembus [8.3] arbeiten die zugeordneten Speichermoduln und Anschlußsteuerungen (AS). Über das auf den Mikrorechnern K2521 installierte Mehrrechnerkopplungsinterface arbeitet die Leitrechner-ZRE als Master mit den ebenfalls aus K2521 realisierten Rechnern für Wegesteuerung PMC (jeweils Slaves) zusammen. Die Mehrrechnerkopplungsinterfaces bilden den Koppelbus, auf dem die Kommunikation der genannten Rechner erfolgt.

Als Speichermodule für die genannten Speicherfunktionen fungieren Steckeinheiten aus dem System K1520 [8.1]:

K3521 4 KByte CMOS-RAM mit Batteriestützung der Information auf der Steckeinheit und Schaltung zur Überwachung der Stützspannung (für Unterprogramme, Korrektur- und Generierdaten, Werkstückprogrammspeicher bis 8 KByte, PMC-Programme während Inbetriebnahmephase);

K3525 16 KByte DRAM mit entsprechenden Bestückungsvarianten 12 KByte und 8 KByte (für Werkstückprogrammspeicher bis 32 KByte);

K3820 16 KByte EPROM; EPROM-Schaltkreise auf Sockel, so daß Bestückungsvarianten in Stufen von 1 KByte möglich sind (Steuerprogrammsystem des Leitrechners, PMC-Programm).

Da der direkt adressierbare Speicherraum des U880 von 64 KByte nicht ausreicht, wird mit Speichererweiterung mittels der Signale MEMDI 1/ und MEMDI 2/ (s. [8.1]) gearbeitet. Die Kommunikation mit der Peripherie erfolgt über Anschlußsteuerungen. Die Steckeinheit AS Bedienung/Leser/Stanzer enthält die Anschlußsteuerungen zum Bedienkomplex der CNC 600, zum eingebauten fotoelektrischen Lochstreifenleser FS 330 (Lesegeschwindigkeit 300 Zeichen/s) und zum Stanzer. Über den Stanzeranschluß sind die Stanzertypen DT 105S, Facit 4070 (Parallelschnittstelle) und daro 1215 (KME3-Pegel und TTL-Pegel) ansteuerbar.

Der Verkehr mit Leser und Stanzer erfolgt nach dem Abfrageprinzip (Polling), mit dem Bedienblendenkomplex nach dem Interruptverfahren. Die AS ANA ist die Anschlußsteuerung für die alphanumerische Anzeige, für die die Bildschirmeinheit ANA 440S (Bildschirmdiagonale 31 cm) verwendet wird. Sie dient als universelle Anzeige in allen Betriebsarten und Modi der Steuerung.

Die AS ANA ermöglicht die Anzeige von 64 beliebigen Zeichen im 5 × 7-Punktraster sowie einer Marke in Doppelstrichform. Die Anzeigeformate 8 Zeilen zu 32 Zeichen bzw. 10 Zeilen zu 48 Zeichen sind über Programm umschaltbar.

Die Steckeinheit PRE/DNC realisiert zwei serielle Schnittstellen mittels U856, die unabhängig voneinander wahlweise als IFSS (*I*nter*f*ace *s*ternförmig *s*eriell) oder als V.24 geschaltet werden können. Die Übertragungsrate ist durch einen Baudratenspeicher einstellbar und kann bei IFSS maximal 9600 Baud, bei V.24 maximal 1200 Baud betragen. Die maximalen Übertragungsentfernungen betragen für IFS 500 m, für V.24 15 m. Auf der Steckeinheit ist außerdem eine Programmspeichererweiterung (PRE) von 4 KByte EPROM untergebracht. Mit dieser Steckeinheit wird der DNC-Betrieb (*d*irect *n*umerical *c*ontrol) mit einem übergeordneten Rechner ausgeführt (On-line-Anschluß). Der EPROM-Programmierzusatz K0420 dient in Verbindung mit dem nachladbaren Programm „Monitor 600" zur EPROM-Programmierung (U855) in der Steuerung.

Die Steckeinheit Überwachung arbeitet als E/A-Baugruppe am Leitrechnerbus und ist in die Interruptprioritätskette eingebunden. Sie erfüllt wichtige Funktionen im On-line- und Off-line-Diagnosesystem der Steuerung CNC 600.

Das On-line-Diagnosesystem der CNC 600 ist hierarchisch gegliedert. Die höchste Priorität haben Hardwareüberwachungen mit direkter Wirkung auf Not-Aus, wobei kein funktionierender Rechner benötigt wird. Dazu gehören die Stromversorgungs- und Taktüberwachung.

In der zweithöchsten Ebene erfolgt eine Hardwareüberwachung der Leitrechnerfunktionsfähigkeit ebenfalls mit direkter Wirkung auf Not-Aus.

In der dritten Ebene laufen die Softwareüberwachungsfunktionen, wobei die Steckeinheit insbesondere Hardwareunterstützung für einige wichtige Funktionen liefert, wie z. B. die Überwachung des richtigen Bussignalspiels am Leitrechnerbus, die Stützspannungsüberwachung der CMOS-Steckeinheiten, die Lüfterüberwachung.

Das System der Softwarediagnose ist bei der CNC 600 weit ausgebaut und bietet hohen Komfort und Sicherheit bei der Handhabung der Steuerung.

Im Rahmen der Off-line-Diagnose stellt die Steckeinheit Überwachung besondere Signale zur Diagnose wichtiger Schnittstellen zur Verfügung.

A1.3. Bedien- und Anzeigekomplex

Der Bedien- und Anzeigekomplex erfüllt im System CNC 600 folgende wesentliche Funktionen:

- Über Bedienelemente (Taster, Drehwahlschalter) und Anzeigen (Tasterrückmeldungen, 7-Segment-Anzeige bei externer Maschinensteuertafel) ermöglicht er die unmittelbaren Bedienhandlungen in allen Betriebsarten und Modi der Steuerung.

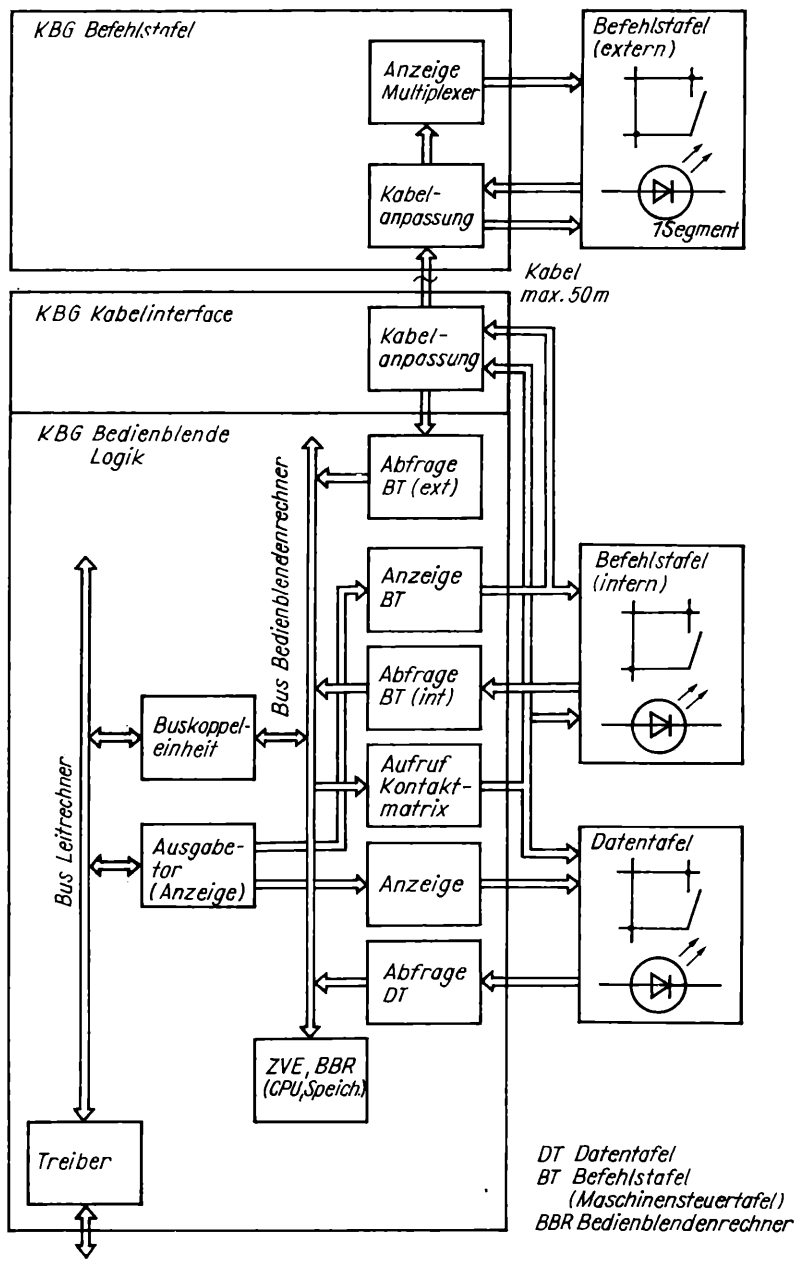


Bild A1.3. Blockschaltbild des Bedien- und Anzeigekomplexes

- Er organisiert die zyklische Zustandsabfrage der Bedienelemente, führt die Prellunterdrückung durch, kodiert die Taster- und Drehwahlschalterzustände und bietet sie dem Leitrechner an.
- Er steuert mit Informationen vom Leitrechner die Anzeigeelemente an (die universelle Bildschirmanzeige gehört nicht zu diesem Komplex).
- Er organisiert Umschaltungen und Verriegelungen.
- Er liefert einen Beitrag zum On-line- und Off-line-Diagnosesystem der Steuerung, so z. B. werden durch ihn ermöglicht

  die zweistufige Temperaturüberwachung,
  die Aufrufbarkeit residenter Diagnoseroutinen im Bedienkomplex durch Leitrechner,
  die Prüfbarkeit des RAM- und EPROM-Bereichs des Bedienblendenrechners durch den Leitrechner.

Das Blockschaltbild des Bedien- und Anzeigekomplexes (Bild A1.3) zeigt die einzelnen Komponenten und deren grundsätzliches Zusammenspiel. Die zentrale Baugruppe dieses Komplexes ist die „Bedienblende Logik", die einen Mikrorechner auf der Basis des Mikroprozessors U880 sowie 0,25 KByte RAM und 1 KByte EPROM enthält. Dieser Rechner arbeitet über die Buskoppeleinheit mit dem Leitrechner auf Interruptbasis zusammen.

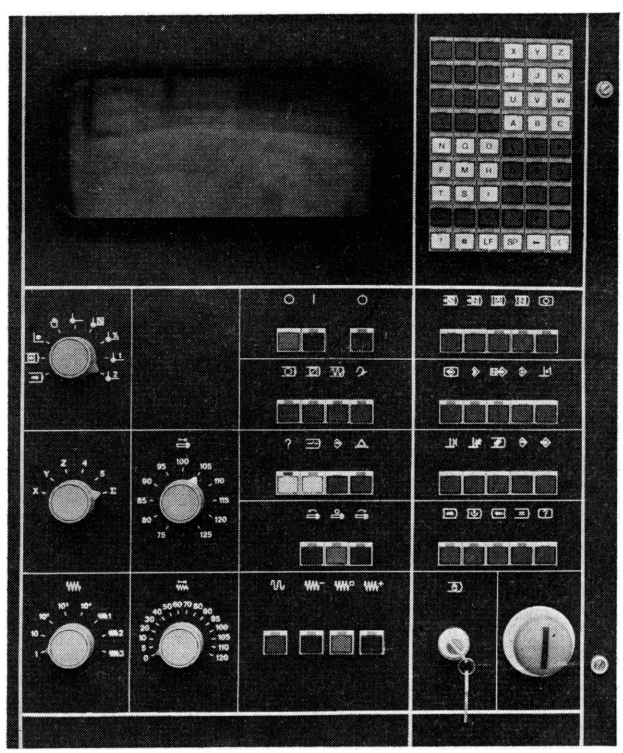


Bild A1.4
Bedienblende CNC 600
(Werkfoto VEB NKM)

Die KGB Kabelinterface wird nur bei Vorhandensein einer externen Maschinensteuertafel benötigt und dient der Überbrückung einer Entfernung $\leqq 50$ m.

Alle Bedien- und Anzeigeelemente sind für den Bediener zu einer Datentafel und einer Maschinensteuertafel zusammengefaßt, die eine übersichtliche, einfache und damit sichere Bedienung ermöglichen.

Während sich die Datentafel immer am Logikschrank befindet, kann die Maschinensteuertafel sowohl am Steuerschrank als auch extern angeordnet werden. Die externe Maschinensteuertafel ist als Einbauvariante gestaltet. Die Kombination sowohl interne als auch externe Maschinensteuertafel ist möglich.

Bild A1.4 zeigt die Bedienblende der CNC 600, bestehend aus Daten und Maschinensteuertafel, für Variante Fräsen.

A1.4. Wegesteuerungs- und Lageregelungskomplex

Die Funktion dieses Komplexes innerhalb des Gesamtsystems besteht darin, die vom Leitrechner aufbereitete und an den Wegesteuerungsrechner übergebene geometrische und Geschwindigkeitsinformation für die auszuführende Bewegung umzuwandeln in Wegsollinformationen, aus denen mit Hilfe der von den Wegmeßsystemen gelieferten Istinformationen für jede Achse die Sollwert-Istwert-Differenz gebildet wird, diese analog zu wandeln und als Führungsgröße für den dem Lageregelkreis unterlagerten Geschwindigkeitsregelkreis vorzugeben.

Dem Teilkomplex Wegesteuerung werden dabei die Aufgaben Grobinterpolation, Ausgaben zu den D/A-Wandlern, Abfrage der Istwertinformationen von den Meßsystemeingängen sowie der Datenaustausch mit Leitrechner und Lageregelungsrechnern übertragen. In jedem der Lageregelungsrechner werden die Aufgaben Feininterpolation, Sollwertglättung, Sollwert-Istwert-Differenzbildung, achsspezifische Korrekturen von Maschinenfehlern (Losekompensation, Spindelsteigungskorrektur) sowie Softwareendschalter und Achsfehlererkennung für eine Achse gelöst.

Die wesentlichen Hardwarekomponenten, auf deren Basis die genannten Funktionen realisiert werden, sind

- ZRE K2521 als Wegesteuerungsrechner, der über das Mehrrechnerkopplungsinterface (Koppelbus 1) mit dem Leitrechner verbunden ist;
- Buskopplung, die die Verbindung zur Peripherie (D/A-Wandler, Meßsystemeingänge) herstellt und den Datenaustausch mit den Lageregelungsrechnern ermöglicht;
- Lageregelungsrechner, der eine autonome Recheneinheit auf der Basis des Mikroprozessors U880 darstellt;
- Meßsystemeingänge (MSE), die die Aufgabe haben, die von den Wegmeßsystemen gelieferten Wegeinformationen zu empfangen und so aufzubereiten, daß der aktuelle Istwert gebildet werden kann; es existieren Meßsystemeingänge sowohl für inkrementale Geber als auch für phasenzyklische Meßsysteme (Inductosyn, Resolver);
- D/A-Wandler, die die digitale Sollwert-Istwert-Differenz einer Breite von 15 bit (einschließlich Vorzeichen) in eine analoge Einheitsspannung im Bereich $-10\ \mathrm{V} \leqq U \leqq +10\ \mathrm{V}$ wandeln.

Die maximale Ausbaustufe ist durch fünf Lageregelungsrechner, fünf D/A-Wandler sowie fünf Meßsystemeingangskanäle gekennzeichnet.

A1.5. PMC-Komplex

Mit dem PMC-Komplex wird die freiprogrammierbare Anpassung an die speziellen Werkzeugmaschinen als integraler Bestandteil des Steuerungssystems CNC 600 gelöst. Die eigentliche Ein-/Ausgabe der Prozeßsignale erfolgt dabei über eine Prozeß-Ein-/Ausgabe-Einrichtung (PEAS), die außerhalb der Steuerung im Leistungsteil der Ma-

schine angeordnet ist. Damit wird eine hohe Flexibilität erreicht, die, unterstützt durch das leistungsfähige Programmsystem der PMC (s. Abschnitte A1.4. und A1.5.), dem Anwender ein effektives Erstellen seiner Programme sowie kurze Inbetriebnahmezeiten ermöglicht. Im unmittelbaren Steuerungsprozeß berechnet die PMC aus vom Leitrechner übergebenen Informationen (NC-Signale) und Zustandssignalen von der Maschine zyklisch nach einem bestimmten Programm Steuersignale für die Werkzeugmaschine (Maschinensignale) und gibt sie über die PEAS aus. In der PMC erfolgt auch die Steuerung des Hauptantriebs (Drehzahl- und Getriebestufenberechnung) sowie zusätzlicher Komplexe (z. B. Werkzeugwechsel, Schmierung u. a.).

Die PMC wird hardwaremäßig realisiert durch eine ZRE K2521, die mit dem Leitrechner über das Mehrrechnerkopplungsinterface verbunden ist (Koppelbus 1). Während das PMC-Steuerprogramm auf der ZRE K2521 untergebracht ist (3 KByte EPROM), steht für das Anwenderprogramm ein Speicherbereich von 16 KByte zur Verfügung. Um in der Inbetriebnahmephase Änderungen mit dem Programm „PMC-Dialog" ausführen zu können, wird ein CMOSRAM (K3521) verwendet. Nach Abschluß der Inbetriebnahme wird das Programm auf EPROM (K3820) übernommen.

Auf einer am Systembus des PMC-Rechners (K1520-Bus) steckenden Anschlußsteuerung (AS PEAS) wird das serienparallele Interface gebildet, das den bidirektionalen Datenverkehr mit der bis zu 50 m entfernten PEAS gerätemäßig abwickelt. Die Übertragungsstrecke ist optisch entkoppelt. Über sie erfolgen Übertragungen mit einer Geschwindigkeit von etwa 20 KByte/s.

A1.6. Rechnerkopplungen

Innerhalb der als Multimikrorechnersystem arbeitenden CNC 600 werden unterschiedliche, den jeweiligen Anforderungen angepaßte Koppelprinzipien zwischen den Mikrorechnern verwendet. Das betrifft die Koppelstellen

- Leitrechner–Bedienblendenrechner,
- Kopplung über Koppelbus 1,
- Kopplung über Koppelbus 2.

Die Kopplung zwischen Leitrechner und Bedienblendenrechner erfolgt über die im Bild 8.8 dargestellte Buskoppeleinheit. Diese stellt eine parallele periphere Schnittstelle auf Basis der Schaltkreise U855 dar, die mit ihrem Businterface am Systembus des Leitrechners angeschaltet sind und vom Bedienblendenrechner erkannte Zustandsänderungen über das Interruptsystem dem Leitrechner melden. Bis zum Abholen der Information durch den Leitrechner verharrt der Bedienblendenrechner im WAIT-Zustand.

Am Koppelbus 1 arbeiten sämtliche K2521-Rechner nach dem Master-Slave-Prinzip, wobei der Master (Leitrechner) mit maximal drei Slaves (Wegesteuerungsrechner, PMC-Rechner, dritter Slaverechner ist bei CNC 600 nicht benutzt) zusammenarbeiten kann. Der Datenaustausch erfolgt zwischen Master und jeweils einem der Slaves bidirektional und kann sowohl vom Master als auch dem Slave initiiert werden. Für die peripherieartige Kopplung benötigt jeder der Rechner einen programmierbaren Ein-/Ausgabe-Schaltkreis mit paralleler Schnittstelle (U855) und eine Zusatzlogik (ZL), die beide auf der Rechnerkarte K2521 installiert sind. Bild A1.5 zeigt die Anordnung der Rechner am Koppelbus 1. Zur Anmeldung eines Sendewunsches bei dem Partner und dessen Quittierung dient der Adressierungs- und Quittungsbus, der aus sechs Leitungen besteht. Für den Fall, daß mehrere Slaves Datenaustausch mit dem Master wünschen, nimmt der

Master die Priorisierung programmgemäß vor. Treffen Mastersendewunsch und Slavesendewunsch aufeinander, dominiert der Master durch Setzen des Masterstatusbits (Übertragungsrichtung). Der eigentliche Datenaustausch erfolgt nach Abschluß des Verbindungsaufbaus in Byteblöcken mit vereinbartem Datenbus. Jedes einzelne Byte wird

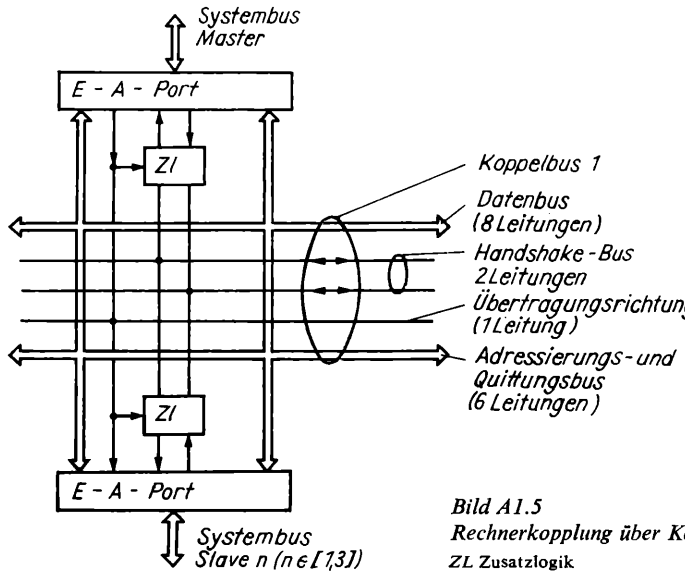


Bild A1.5
Rechnerkopplung über Koppelbus 1
ZL Zusatzlogik

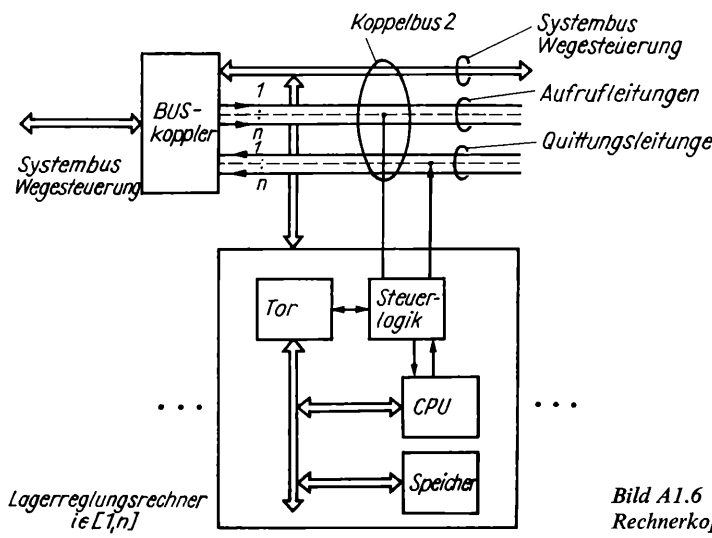


Bild A1.6
Rechnerkopplung über Koppelbus 2

dabei im asynchronen Handshake (Handshakebus) vom Sender zum Empfänger übergeben. Die Zusammenarbeit zwischen der peripheren Schnittstelle und der jeweiligen CPU erfolgt auf Interruptbasis. Die Datenübertragung auf dem Koppelbus 1 erfolgt mit einer Geschwindigkeit von etwa 30 KByte/s. In die Rechnerkopplungsprogramme sind eine Reihe von Überwachungsfunktionen implementiert, die im Fehlerfall zu Fehleranzeigen am Bildschirm führen.

Die Kopplung von Wegesteuerungs- und maximal fünf Lageregelungsrechnern erfolgt

über Koppelbus 2 (Bild A1.6). Im normalen Betrieb arbeiten alle beteiligten Rechner autonom. Der Datenaustausch wird stets vom Wegesteuerungsrechner (Master) eingeleitet. Zu diesem Zweck enthält Koppelbus 2 neben den Signalen des Wegesteuerungssystembusses Aufruf- und Quittungsleitungen. Bei Aufruf eines der Lageregelungsrechner trennt sich dessen CPU vom lokalen Bus und quittiert dies. Gleichzeitig wird durch Öffnen des Tores der Wegesteuerungsbus auf den Lageregelungsrechner geschaltet und der Datentransfer als Speichertransfer der Wegesteuerung durchgeführt. Zu beachten ist dabei, daß sich die Adreßräume von Wegesteuerungs- und Lageregelungsrechner nicht überlappen. Bei der beschriebenen Art des Datenaustausches wird eine Übertragungsgeschwindigkeit von mehr als 100 KByte/s erreicht.

A1.7. Prozeß-Ein-Ausgabe-Steuerung

Die Prozeß-Ein-/Ausgabe-Steuerung (PEAS) kann als konstruktiv selbständiger Komplex, bestehend aus Logikteil, Stromversorgung und Lüfterbaugruppe außerhalb des CNC-600-Steuerschranks (z. B. im Leistungsteil der zu steuernden Maschine) untergebracht sein. Funktionell dagegen ist sie nicht selbständig, sondern arbeitet mit dem PMC-Komplex zusammen.

Die Hauptaufgaben der PEAS sind die Organisierung der Verbindung zur PMC, die nach Kanälen geordnete Durchschaltung der Eingangs- und Ausgangssignale, die Verstärkung der Ausgangssignale auf das erforderliche Niveau zur direkten Ansteuerung von Stellgliedern und die Anpassung der Eingangssignale an die internen Pegel. Darüber hinaus enthält die PEAS Überwachungseinrichtungen, die die Steuerung bei auftretenden Fehlern in den für die Maschine und den Bediener ungefährlichen Zustand bringen. Diagnoseeinrichtungen ermöglichen unter Zuhilfenahme von Prüfprogrammen (in PMC) das Auffinden der defekten Steckeinheit.

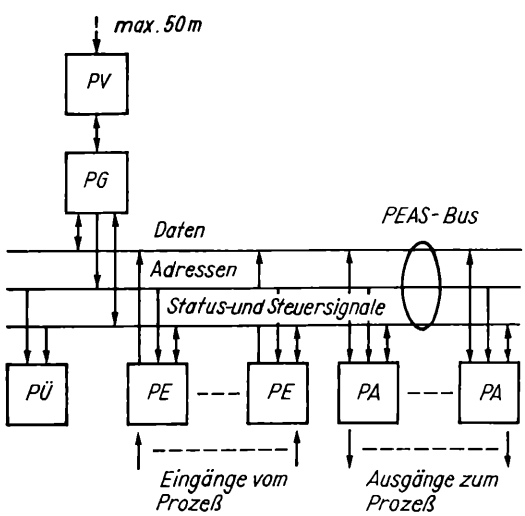


Bild A1.7
Blockschaltbild der PEAS

Das Blockschaltbild der PEAS ist im Bild A1.7 dargestellt. Verbindungsbaugruppe PV, Grundbaugruppe PG und Überwachungsbaugruppe PÜ müssen als Grundausrüstung stets vorhanden sein. Zahl und Art der Eingabebaugruppen PE und Ausgabebaugruppen PA sind dagegen projektierbar. Vom Programm verarbeitbar und adressierbar sind maximal 480 Prozeßeingangs- bzw. -ausgangssignale. Konstruktiv stehen

20 Steckplätze bei einer Logikkassette bzw. 40 Steckplätze bei zwei Logikkassetten für Prozeßeingabe- und -ausgabekarten zur Verfügung.

Die Stromversorgung der PEAS liefert alle zum Betreiben der PEAS-Steckeinheiten notwendigen Spannungen einschließlich der Versorgung der Eingabeelemente im 24-V-Pegel. Die Versorgung der Stellglieder und der Eingabeelemente für 110 V Ws ist vom Anwender bereitzustellen.

Die Logik der PEAS ist in CMOS-Technik ausgeführt.

Jeder Ein-/Ausgabe-Modul der PEAS besitzt eine Einzelbitanzeige mittels Lumineszenzdioden und einen steckbaren Einzelbitanschluß.

A1.8. Software des Steuerungssystems CNC 600

Die Software des Systems CNC 600 gliedert sich in

- Steuerprogrammsystem,
- Anwenderprogramme,
- Serviceprogrammsystem.

Diese Programme sind den Speicherbausteinen der einzelnen Mikrorechner der Steuerung zugeordnet.

Das *Steuerprogrammsystem* realisiert den NC-spezifischen Funktionsinhalt der Steuerung (Funktionsprogramme) und ist auf EPROM abgelegt (residente Programme). Zur Abarbeitung der einzelnen Steuerprogramme werden außerdem in jedem Mikrorechner interne Arbeitsspeicher und Datenfelder als RAM-Bereiche benötigt.

Die *Anwenderprogramme* stellen die abzuarbeitenden Eingabedaten für die Steuerprogramme dar. Sie gliedern sich in die Werkstückbearbeitungsprogramme (geometrische und technologische Daten für das zu bearbeitende Werkstück) und die Anpaßprogramme (Beschreibung der Signalverknüpfung an der Schnittstelle Steuerung–Maschine). Die Anwenderprogramme werden je nach Ausführungsvariante der Steuerung in dynamische oder batteriegestützte statische RAM (CMOS-RAM) eingelesen. Das Anpaßprogramm kann auch auf EPROM programmiert werden.

Das *Serviceprogrammsystem* dient zur Unterstützung von Inbetriebnahme-, Wartungs- und Fehlersucharbeiten. Durch Doppelausnutzung von RAM-Bereichen ist eine Vielzahl von Serviceprogrammen einzeln nachladbar (nachladbare Programme). Die Serviceprogramme belegen nach dem Einlesen einen Teil der RAM-Speicher, in die sonst Werkstückbearbeitungsprogramme eingelesen werden. Das hat zur Folge, daß in diesem Fall der normale NC-Betrieb der Steuerung nicht oder nur eingeschränkt möglich ist.

A1.9. Steuerprogrammsystem

Jeder Rechner des Multimikrorechnersystems der Steuerung realisiert durch die ihm fest zugeordneten Steuerprogramme bestimmte Funktionskomplexe.

Jeder Rechner arbeitet seine Funktionsprogramme autonom ab. Leitrechner, Wegesteuerungs- und PMC-Rechner können untereinander programmierten Datenaustausch durchführen, während der Wegesteuerungsrechner durch DMA (direkter Speicherzugriff) zu den Speichern der Lageregelungsrechner Zugriff hat. Der Programmablauf wird in jedem Rechner eigenständig und nach einem der jeweiligen Funktion angepaßten Prinzip organisiert.

Der *Leitrechner* arbeitet nach dem Interruptprinzip. Die Meldungen der Slaves sowie

die des Bedienblendenrechners (Bedieninformationen) und einige Fehlersignale des Überwachungssystems werden über Interruptprogramme erfaßt und ausgewertet. Die daraufhin notwendigen Reaktionsprogramme zur Informationsverarbeitung und zum Informationstransport werden von den Interruptprogrammen in drei Wertigkeitsebenen angemeldet und nach dem Warteschlangenprinzip abgearbeitet.

Wichtige Leitrechnerprogrammkomplexe werden nachfolgend kurz erläutert.

Das *Betriebssystem* organisiert den Steuerprogrammablauf im Leitrechner, wertet Fehlersignale der Überwachungsbaugruppe aus, steuert den Datenaustausch mit Wegesteuerung und PMC, mit Leser und Stanzer und nimmt die Einschaltinitialisierung vor.

Das Programm *Grunddiagnose* diagnostiziert nach dem Einschalten den Hardcore des Leitrechners und gibt ggf. Fehlermeldungen im Klartext über den Bildschirm aus.

Der Programmkomplex *Geometrieberechnung* stellt eine Unterprogrammbibliothek zur Schnittpunktberechnung geometrischer Formelelemente dar. Diese Formelemente werden beim Einlesen des Werkstückprogramms aufgelöst oder müssen zur Definition der äquidistanten Bahnkurve bei Fräserradiuskorrektur berechnet werden.

Die *Standardunterprogramme* stellen eine Programmbibliothek für arithmetische Berechnungen im Fest- und Gleitkommaformat bzw. für die Konvertierung von Wortformaten dar.

Das Programm *Maschinensteuertafel* realisiert die Bedienfunktionen der Maschinensteuertafel einschließlich der Bildschirmanzeigen in den einzelnen Betriebsarten der Steuerung.

Das Programm *Ablaufsteuerung* wertet die Meldungen von Wegesteuerung und PMC aus und führt die entsprechenden Reaktionen einschließlich der Ausgaben an Wegesteuerung und PMC aus beziehungsweise meldet sie an (Satzaufbereitung).

Im Programm *Satzaufbereitung* werden das Werkstückbearbeitungsprogramm satzweise dekodiert und die internen Satzpuffer gefüllt.

Im Programm *NC-Vorbereitung* werden die internen Satzpufferinformationen zur Ausgabe vorbereitet, indem die Fräserradiuskorrektur über die Geometrieberechnung in die Bezugsmaße eingerechnet und die Vorschubkonstante gebildet wird.

Informationen für die Wegesteuerung werden ausgegeben.

Das Programm *Datentafel* erledigt alle Bedienfunktionen der Datentafel einschließlich zugehöriger Bildschirmanzeigen.

Das *Einleseprogramm* realisiert die Eingabe, Syntaxkontrolle und Abspeicherung der Werkstückbearbeitungsprogramme über Lochstreifenleser oder Tastatureingabe und ruft erforderlichenfalls die Geometrieberechnung auf.

Das Programm *Istwerterfassung* wertet zyklisch die Wegistwerte der Wegesteuerung aus zur Anzeige, Achsfehlererfassung sowie für die drehspezifischen Funktionen Umdrehungsvorschub und konstante Schnittgeschwindigkeit.

Das Programm *Schnittaufteilung* realisiert die automatische Schnittaufteilung für die Technologie Drehen, indem in Abhängigkeit von den Istkoordinaten Verfahrbewegungen generiert werden.

Das *Steuerprogramm der PMC* arbeitet zyklisch die Interpretation der Booleschen Gleichungen und die Assemblerprogramme ab und führt die Abfrage der Eingänge und das Setzen der Ausgänge aus. Dieser Zyklus wird nur unterbrochen von den Interruptprogrammen der Rechnerkopplung oder von einem kurzen CTC-Interruptprogramm zur Realisierung der Zeitglieder. Als Assemblerprogramme sind die Programme für Werkzeugwechsel, Spindelsteuerung und Schmierung realisiert.

Aufgrund der zyklischen Arbeitsweise ergibt sich eine vom Steuerungsumfang abhängige Zykluszeit der PMC (etwa 10 ms je 1 KByte Anwenderprogramm).

Das *Steuerprogrammsystem der Wegesteuerung* untergliedert sich in Hinter- und Vordergrundprogramme. Während der Abarbeitung der zur Informationsaufbereitung ständig ablaufenden Hintergrundprogramme unterbricht ein 2,5 ms-Takt (CTC-Interrupt) diese Abarbeitung regelmäßig und stößt zyklisch verschiedene Zweige des Vordergrundprogramms an, das die Ein- und Ausgaben mit den Meßsystemeingängen bzw. D/A-

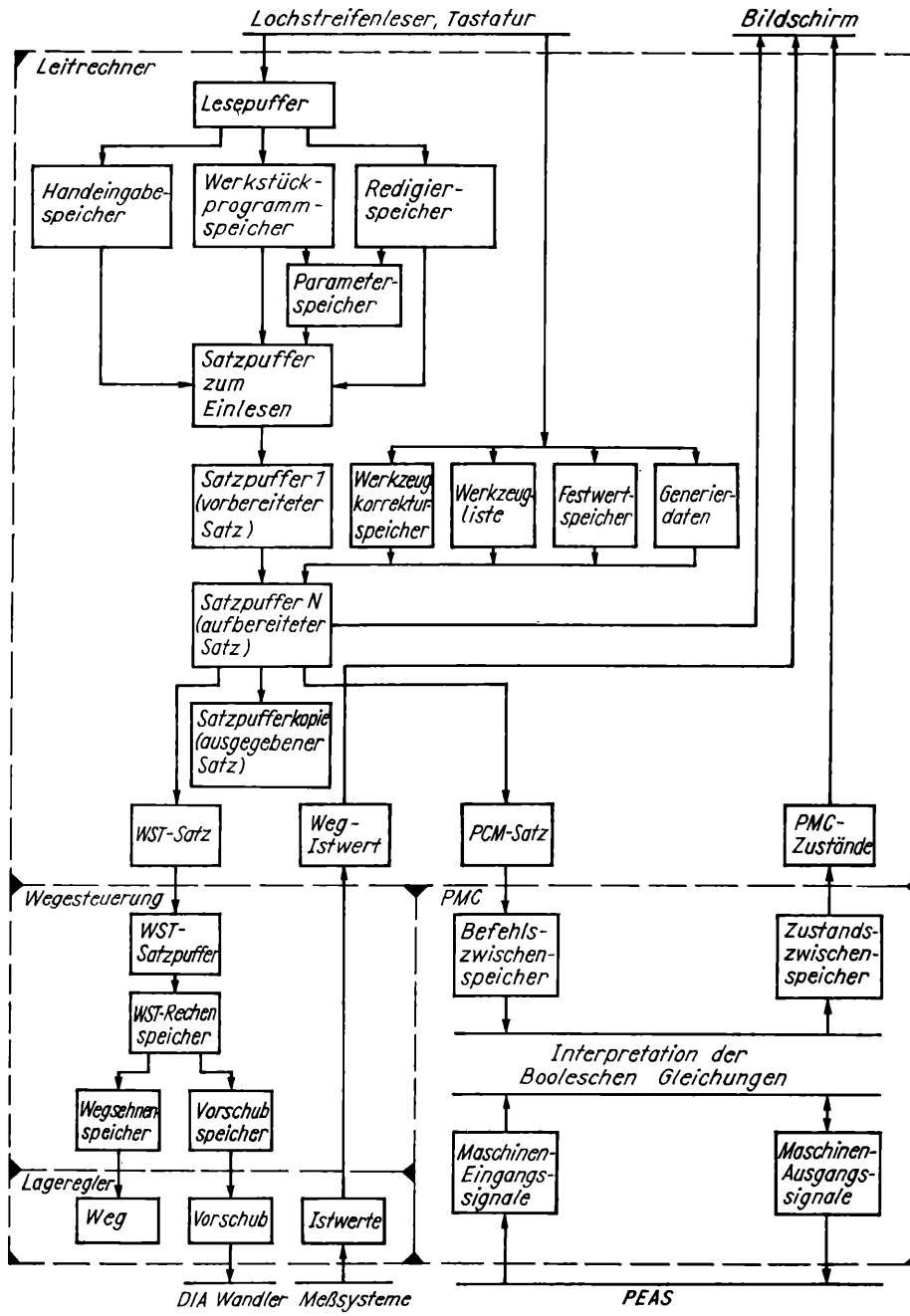


Bild A1.8. Informationsfluß in der CNC 600

Wandlern realisiert. Die Interruptprogramme der Rechnerkopplung zum Leitrechner unterbrechen den Hintergrund, werden aber selbst vom Vordergrundprogramm unterbrochen.

Das *Steuerprogramm Lageregelungsrechner* besteht aus den Teilen Sollwertberechnung und Sollwert-Istwert-Vergleich, die jeweils taktsynchron vom Vordergrundprogramm der Wegesteuerung gestartet werden und in einer Warteschleife enden.

Das *Programm des Bedienblendenrechners* läuft zur Abfrage der Taster- und Wahlschalterstellungen bzw. zur Meldung von Kodierungen zyklisch ab.

Der durch das Steuerprogrammsystem erzeugte Informationsfluß ist schematisch im Bild A1.8 dargestellt. Er enthält alle wichtigen Speicherbereiche, in denen NC-Daten gespeichert bzw. aufbereitet werden.

Steuerinformationsspeicher sind der Übersichtlichkeit halber nicht dargestellt. Steuersignale, die die Informationsverarbeitung bzw. den Informationstransport auslösen, entstehen bei Bedienhandlungen, Prozeßzustandsänderungen, bei Auswertung bestimmter Daten oder durch Rechnerkopplung.

A1.10. Serviceprogrammsystem

Das Serviceprogrammsystem besteht aus den Teilsystemen

- Monitorprogramm,
- Generierungsprogramm,
- Diagnoseprogramm,
- PMC-Dialog,
- PMC-Dokumentation,

die nachfolgend kurz bezüglich ihres Inhalts erläutert werden.

Das *Monitorprogramm* dient dem Anwender vor allem zum Duplizieren von Lochstreifen, zum Lochstreifenvergleich mit EPROM-Inhalten sowie in Verbindung mit dem EPROM-Programmierzusatz K0420 zum Programmieren von EPROM. Außerdem ist mit diesem Programm das Anzeigen von Speicherinhalten sowie das Ausstanzen und Wiedereinlesen des Anpaßprogramms der PMC im Maschinenkode möglich.

Das *Generierungsprogramm* dient bei der Inbetriebnahme der Steuerung dazu, steuerungs- und maschinenspezifische Daten (Generierungsdaten) eingeben bzw. ändern zu können. Nach der Generierung der Steuerung werden die neuen Generierdaten (in der Steuerung auf gestütztem CMOS-Speicher abgelegt) durch Ausstanzen fixiert.

Das Diagnoseprogrammsystem besteht aus einem Diagnosebetriebssystem und einer Menge von Testroutinen zur Diagnostizierung der Steuerungsbaugruppen. Die Diagnoseroutinen können einzeln oder nacheinander in einer logischen Reihenfolge eingelesen werden. Sie werden unter Steuerung des Diagnosebetriebssystems abgearbeitet. Das simultane Abarbeiten von Werkstückprogrammen ist nicht möglich. Dieses Off-line-Diagnosesystem stellt neben den On-line-Überwachungen und -diagnosen und dem Einschaltdiagnoseprogramm einen wesentlichen Teil des gesamten Diagnose- und Überwachungssystems der CNC 600 dar. Es dient insbesondere im Fehlerfall zu einer schnellen Lokalisierung der fehlerhaften Baugruppe.

Mit Hilfe des *PMC-Dialogs* läßt sich das vom Anwender erstellte Anpaßprogramm in die PMC einlesen. Dabei erfolgen Syntaxkontrolle und Konvertierung. Auftretende Fehler sowohl beim Einlesen als auch beim Abarbeiten des PMC-Programms lassen sich durch Zustandsanzeigen orten und über Kommandos korrigieren. Der PMC-Dialog

existiert zusätzlich in einer abgerüsteten Version für Servicezwecke. Die Anwendung des PMC-Dialogs bei der Inbetriebnahme eines PMC-Programms geht aus Bild A1.9 hervor.

Mit dem *Programmsystem zur PMC-Dokumentation* können über Lochstreifen und Schreibautomat zwei Unterlagen erstellt werden. Das Programm Zustandsgraphen stanzt einen Lochstreifen aus, der beim Ausdrucken auf einem Schreibautomaten eine schematische Darstellung der Verknüpfungen des PMC-Programms in Form von Signalzustandsgraphen liefert.

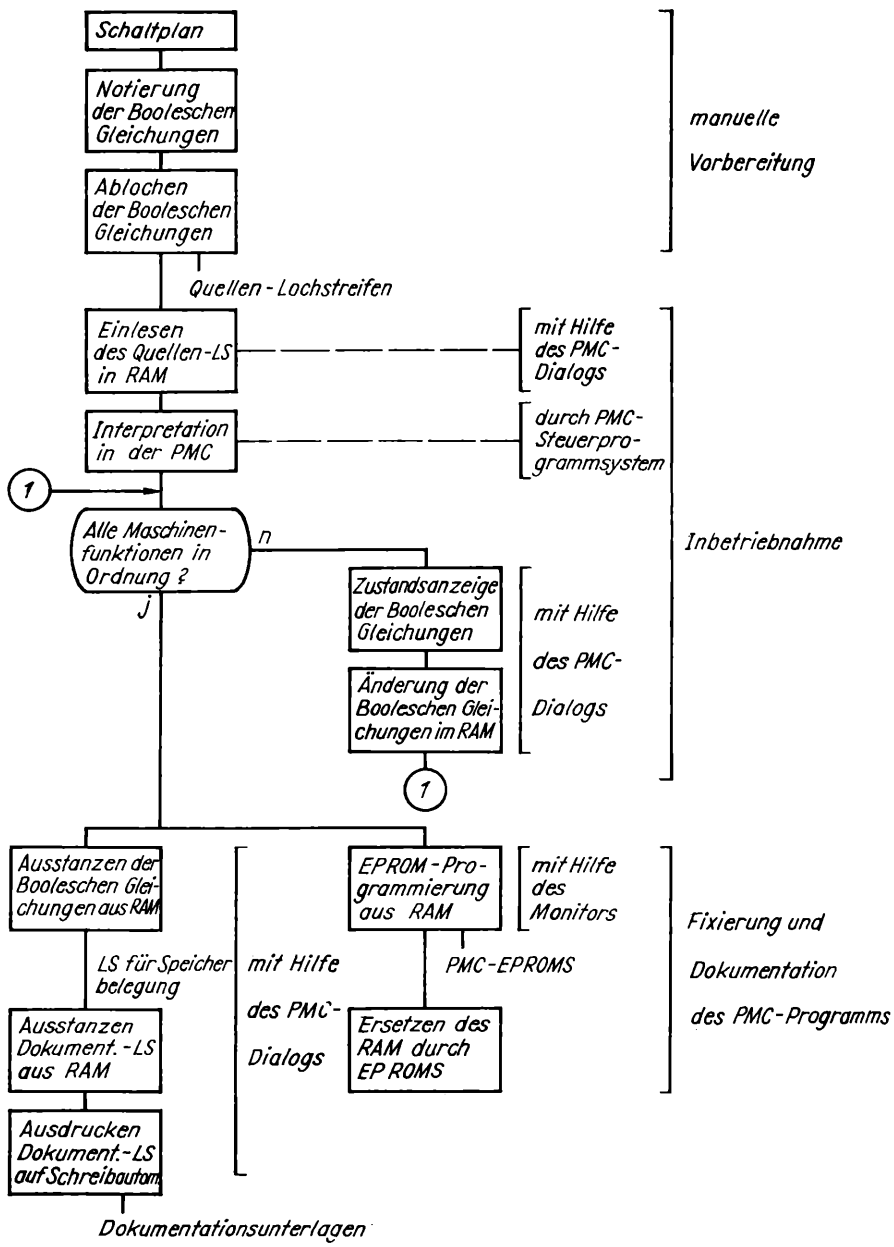


Bild A1.9. Schritte bei der Inbetriebnahme eines PMC-Programms

Das Programm Cross-Referenz liefert ebenfalls einen Lochstreifen, mit dem eine Cross-Referenz-Liste aller im PMC-Programm vorkommenden Variablen ausgedruckt werden kann.

A1.11. Programmierung

Die Erstellung der Anwenderprogramme für das zu bearbeitende Werkstück (Werkstückprogrammierung) und für die Anpassung der Steuerung an einen bestimmten Werkzeugmaschinentyp (PMC-Programmierung) hat nach den entsprechenden Programmieranleitungen [8.8] zu erfolgen. Dennoch sollen für das allgemeine Verständnis einige Aspekte der Programmierung dargelegt werden. Als Lochstreifencode dient der ISO-7-bit-Kode nach TGL 23207/01 bzw. der Kode E/A RS 244. Die Kodeerkennung erfolgt automatisch in der Steuerung anhand des Programmanfangszeichens. Aus den Zeichen eines festgelegten Zeichenvorrats werden Wörter definierten Formats gebildet. Der Satzaufbau erfolgt gemäß TGL 200-0863/02, die maximale Satzlänge beträgt 96 Zeichen. Viele Sätze ergeben ein Programm. Zur besseren Übersichtlichkeit von Programmen kann die Hauptsatzschreibweise angewendet werden. Einige für die neue Steuerungsgeneration typische Programmierfunktionen, die dann auch den Funktionsinhalt der Steuerung charakterisieren und durch das Steuerprogrammsystem realisiert werden, seien nachfolgend angeführt.

Äquidistantenberechnung

Die Steuerung führt eine vollständige Äquidistantenberechnung aus sowohl für Innen- als auch für Außenkonturen. Ebenenangaben sind nicht erforderlich.

Parameterzuweisung in NC-Programmen und in Unterprogrammen

NC-Adressen können neben festen Zahlenwerten Parameter zugewiesen werden. Vor ihrer Zuweisung müssen sie grundsätzlich definiert werden.

Rechenvorschriften für Parameter

Neben der direkten Zahlenzuweisung ist es möglich, Parameter aus Parametern und Zahlenwerten zu errechnen. Dabei sind die Rechenvorschriften +, −, ·, :, sin, cos zulässig.

Unterprogrammtechnik

Unterprogramme (UP) sind für mehrfach nutzbare Programmteile in Werkstückprogrammen empfehlenswert. Das UP wird nach Abarbeitung des NC-Satzes ausgeführt, der den UP-Aufruf enthält. Unterprogramme können wieder Unterprogramme aufrufen; dabei sind maximal fünf Unterprogrammebenen erlaubt.

Es wird zwischen allgemeinen technologischen Unterprogrammen (z.B. Bohrzyklen, Lochmuster u.a.) und hauptprogrammspezifischen Unterprogrammen (zur Vereinfachung des Hauptprogramms) unterschieden. Erstere werden im Unterprogrammspeicher abgelegt und stehen für jedes beliebige NC-Programm zur Verfügung.

Werkstückspezifische UP werden mit im Werkstückprogrammspeicher abgelegt und stehen nur für das dafür vorgesehene Werkstückprogramm zur Verfügung.

Programmabschnittswiederholung

Mittels dieser Funktion können im Hauptprogramm und in den Unterprogrammen Programmschleifen definiert werden. Die Auswahl der Durchläufe einer Schleife wird mit

einem zweistelligen Wort gekennzeichnet. Programmabschnittswiederholungen sind in maximal fünf Ebenen schachtelbar. Innerhalb von Programmabschnitten können Unterprogramme aufgerufen werden.

Geometrische Formelelemente

Mit Hilfe dieser Funktion können Punkte, Geraden und Kreise definiert und durch die Steuerung berechnet werden, die im weiteren Programmablauf benötigt werden, jedoch nicht aus der Werkstückzeichnung ablesbar sind. Die Funktion „geometrische Formelelemente" stellt somit ein Hilfsmittel dar, das zur Berechnung der verschiedenen geometrischen Probleme ohne Zuhilfenahme externer Rechner verwendet wird. Die hauptsächlichen Anwendungsgebiete sind die Konturbeschreibung der Werkzeugbahn und Bohrbildbeschreibungen. Geometrische Formelelemente können in Form von sog. Rechensätzen an beliebiger Stelle im Werkstückprogramm geschrieben werden. Für diese Rechensätze ist ein spezielles Format zu verwenden. Sie definieren (u. U. in mehreren Schritten) einen bestimmten Punkt, dessen Koordinaten unter den im Satz geschriebenen Q-Adressen im weiteren Programmablauf anwählbar sind. Im eigentlichen Satz des Teileprogramms wird der jeweiligen Koordinate der *vorher* im Rechensatz berechnete konkrete Zahlenwert zugewiesen. Die durch die Rechensätze angewiesenen Rechnungen werden während der Einlesephase ausgeführt, so daß die Rechensätze nicht den Werkstückprogrammspeicher belasten.

Weitere Programmierfunktionen stehen für die Werkzeugwahl und die Autoprogrammierung beim Drehen zur Verfügung.

Die Steuerung erkennt eine Vielzahl von Programmierfehlern, die einzelnen Funktionen zugeordnet sind und in kodierter Form am Bildschirm angezeigt werden. Sie gewährleistet damit in Verbindung mit den komfortablen Programmredigiermöglichkeiten vor Ort über Datentafel und Bildschirm eine effektive Inbetriebnahme von Werkstückprogrammen.

PMC-Programmierung

Die Darstellung in Form von Booleschen Ausdrücken, die aus logischen Verknüpfungen von Eingangsvariablen, Ausgangsvariablen, Zwischenspeichern u. ä. bestehen, stellt den Kern für die Programmierung der PMC 600 dar. Mittels dieser Booleschen Ausdrücke werden logische Wertezuweisungen (Boolesche Gleichungen), bedingte Sprünge, Fehlermeldungen und Anzeigen notiert. Dieser Teil der Programmierung kann durch Einfügen von Assemblerprogrammen universell erweitert werden. Der Logikteil des PMC-Programms wird in problemangepaßter Form unabhängig von dem verwendeten Mikroprozessor notiert, die Assemblerprogramme sind im Kode des Mikroprozessors U880 zu schreiben.

Die Erfassung der PMC-Programme erfolgt im ISO-7-bit-Kode oder im EIA-Kode. Für das Formulieren der Booleschen Gleichungen ist es dabei gleichgültig, ob die notwendigen Informationen aus einem Impulsdiagramm, einem Relaisplan, mittels automatentheoretischer Methoden oder aus einer verbalen Beschreibung der Steuerungsaufgabe gewonnen werden.

Zur Bildung der Booleschen Ausdrücke stehen die Operatoren UND, ODER, NICHT sowie öffnende und schließende Klammern zur Verfügung. Innerhalb eines solchen Ausdrucks können 1 bis 42 Variable verknüpft werden. Mehrfache Klammerung ist zugelassen; das Klammerniveau darf niemals negativ werden.

Durch eine logische Gleichung wird ein Boolescher Ausdruck einer Variablen zu-

geordnet. Als Variable fungieren Eingänge, Ausgänge, Zwischenspeicher, Zeitgliedeingänge und -ausgänge sowie NC-Befehle. Von den 480 Zwischenspeichern der PMC 600 sind 300 für die Anwenderlogik frei verfügbar.

Jede Boolesche Gleichung beginnt mit einer Satznummer, die eine Bezugnahme auf die Gleichung bei bedingten Sprüngen bzw. beim PMC-Dialog ermöglicht. Ein bedingter Sprung wird z. B. in der Form geschrieben:

Nxxx : Nyyy = Boolescher Ausdruck

Dabei ist Nxxx die laufende Satznummer. Ist der Boolesche Ausdruck wahr, wird das Programm bei Nyyy fortgesetzt; ist er nicht wahr, erfolgt die Fortsetzung bei dem auf Nxxx folgenden Satz. Sprünge werden vor allem benutzt, um die Abarbeitung der Steuerlogik zu beschleunigen.

Eine wichtige Sonderfunktion, die im Rahmen der PMC-Programmierung zur Verfügung gestellt wird, ist die Anzeigemöglichkeit von bis zu 199 Maschinenfehlern in folgender Art:

Nxxx : Fnnn = Boolescher Ausdruck

Ist der Boolesche Ausdruck wahr (in ihm kann der Anwender bestimmte Maschinenzustände verknüpfen), so wird Fnnn am Bildschirm der CNC 600 angezeigt. Liegt der Zustand „falsch" vor, so erfolgt keine Anzeige.

Eine weitere programmierbare Funktion sind die sog. Anzeigebits, mit denen wichtige Zustände zur Anzeige am CNC-Bildschirm gebracht werden können. Zu diesem Zweck sind 16 Zwischenspeicher Z reserviert. Die Programmierung erfolgt in der Form

Nxxx : Zyy = Boolescher Ausdruck

Eine Anzeige erfolgt, wenn der Boolesche Ausdruck wahr ist.

Zur Realisierung von Funktionen, die sich in Boolescher Algebra nur umständlich ausdrücken lassen, können in die Logik Programme im U880-Kode (Assemblerprogramme) eingefügt werden:

Nxxx : B = (Programm im U880-Kode)

Die Assemblerprogramme können auf alle Variablen zugreifen und somit z. B. auch Zwischenspeicher beeinflussen, die in Booleschen Gleichungen weiterverarbeitet werden. Bereits im Betriebssystem der PMC 600 sind die Assemblerprogramme

- Zeitgliedroutine,
- Hauptantriebssteuerung,
- Positionierroutine für Werkzeugwechsel

enthalten.

Die Zeitgliedroutine behandelt 32 Zeitglieder. Zeitglieder können programmiert werden in Schritten von 0,1 s von 0 bis 99 min 59,9 s.

Das Programm Hauptantriebssteuerung hat als Hauptaufgaben die Bestimmung der Getriebestufe, der Sollwertspannung für den D/A-Wandler, das Einarbeiten des Override sowie die Freigabe der Sollwertspannung unter Berücksichtigung der Drehrichtung. Die Positionierroutine für Werkzeugwechsel erfüllt für einen inkremental arbeitenden Werkzeugspeicher die Funktionen Referenzpunktanfahren, Ordnung der Speicherbestückung, Speicherpositionierung auf Sollwert und Ablaufüberwachung.

A1.12. Freiprogrammierbare Steuerung für automatische Fertigungszellen CNC 603

Flexible Fertigungszellen werden bei der weiteren Rationalisierung der spanenden Fertigung von Klein- und Mittelserien eingesetzt. Sie sind dadurch gekennzeichnet, daß ein meist mehrere Bearbeitungsarten erforderndes Teilespektrum weitestgehend automatisiert im bedienerarmen bzw. bedienerlosen Betrieb gefertigt wird.

Zur Steuerung derartiger Fertigungszellen für den technologischen Prozeß Bohren/Fräsen wurde die Steuerung CNC 603 entwickelt, die gegenüber der üblichen CNC 600 eine Reihe wesentlicher Funktionserweiterungen enthält. Dazu gehören insbesondere:

- Automatikbetrieb für bedienerarmen bzw. bedienerlosen Betrieb,
- Werkstückkontrolle,
- Werkzeugkontrolle,
- elektronisches Gewindeschneiden und Spindelrichten,
- erweiterter Funktionsinhalt bei der NC-Programmierung,
- Prozeßdatenerfassung,
- wegabhängige Schmierung,
- analoge Meßwerteingabe in die PMC (über PEAS).

Die Erweiterungen im *Automatikbetrieb* beziehen sich auf die Listenführung bei vorhandenem Werkstückspeicher (Zuordnung Werkstückprogramm–Palettennummer) und auf die zwei realisierten Betriebsarten für die Auswahl des zu bearbeitenden Werkstücks. In der ersten Betriebsart erfolgt der Palettenaufruf durch das PMC-Programm, und das zugehörige Werkstückprogramm wird automatisch gestartet.

In der zweiten kann die Programmreihenfolge vorgegeben werden, und es erfolgt automatisch das Einwechseln der entsprechenden Palette.

Für die *Werkstückkontrolle* dient ein 3-D-Meßtaster zur Erfassung der Werkstückkonturen in Richtung der translatorischen Achsen. Die ermittelten Meßwerte werden als Parameter (im erweiterten Parameterspeicher) achszugeordnet gespeichert und im NC-Programm verrechnet. Auf diese Weise kann sowohl das automatische Einrichten eingewechselter Werkstücke als auch die Kontrolle der Maßhaltigkeit eines Werkstücks während bzw. nach der Bearbeitung erfolgen. Mittels eines Kalibertasters kann die Kontrolle von Bohrungen auf Maßhaltigkeit erfolgen. Sowohl 3-D- als auch Kalibertaster werden von der Steuerung wie ein Werkzeug im Werkzeugmagazin behandelt; ihre Maßwerte werden telemetrisch zur Steuerung übertragen.

Die *Werkzeugkontrolle* umfaßt die Standzeitüberwachung der Werkzeuge, die optische Erfassung von Werkzeugbruch mittels einer Fernsehkamera sowie die Möglichkeit einer automatischen Fortsetzung der Bearbeitung bei gebrochenem bzw. verschlissenem Werkzeug mit Ersatzwerkzeugen.

Das *elektronische Gewindeschneiden und Spindelrichten* erfolgt durch Ausrüstung der Hauptspindel mit einem Meßsystem. Die Hauptspindel wird so als fünfte NC-Achse lagegeregelt.

Zum *erweiterten Funktionsinhalt bei der NC-Programmierung* sind insbesondere zu rechnen:

- 6 Festwerte je Achse,
- 200 Parameterspeicherplätze,
- erweiterte Rechenmöglichkeiten ($\sqrt{}$, tan, arctan, arcsin),
- vereinfachte Verarbeitung negativer Parameter,
- Korrektur der Festwerte und Werkzeugkorrekturwerte durch NC-Programme (z. B. Meßroutinen).

Im Rahmen der *Prozeßdatenerfassung* wird ein Protokoll der Maschinenauslastung erstellt, das z. B. Uhrzeit (mit Echtzeituhr realisiert), Stückzeit, Einsatzzeit, bearbeitete Werkstücke je Programm usw. enthält.

Zur Realisierung der wegabhängigen Schmierung werden die verfahrenen Wege je Achse erfaßt, und bei Erreichen der für die Schmierung generierten Wegspanne wird über PMC/PEAS ein entsprechender Schmierimpuls ausgegeben.

Die in der PMC/PEAS realisierte *analoge Meßwerteingabe* kann zur Erfassung kritischer Maschinenzustände im Rahmen einer Maschinendiagnose genutzt werden (z. B. Erfassung von Lagertemperaturen, Ständerverzerrungen u. a.). Es stehen acht analoge multiplexe Eingangskanäle einer Wandlungsbreite von 8 bit zur Verfügung.

Für die Realisierung der erläuterten Zusatzaufgaben werden, ausgehend von der CNC 601 – Ausbaustufe für fünf Achsen und 32 KByte Werkstückprogrammspeicher folgende Zusatzbaugruppen benötigt:

- Erweiterung des Speicherausbaus im Leitrechner um 4 KByte CMOS-RAM, 16 KByte EPROM und 32 KByte DRAM für Werkstückprogrammerweiterung mit den Speichermoduln K3521, K3820 und K3525 unter Ausnutzung der bereits vorbereiteten Speichererweiterung;
- Steckeinheit „schneller" Eingang zur Erfassung und Bereitstellung des Meßtastersignals, die am Wegesteuerungsbus arbeitet;
- Steckeinheit Werkzeugbrucherkennung, die von der Fernsehkamera kommende Signale auswertet und am PEAS-Bus arbeitet;
- Steckeinheit zur A/D-Wandlung für acht multiplex aufrufbare Kanäle mit einer Wandlungsbreite von 8 bit (Eingangsspannungsbereich -5 bis $+5$ V, Wandlungszeit 40 μs), diese Steckeinheit arbeitet am PEAS-Bus;
- Steckeinheit, die gestattet, vier analoge Widerstandsmeßwandler anzuschließen und die die Anpassung an die genannte Steckeinheit zur A/D-Wandlung durchführt.

A2. Gerätesystem ursalog 5000

Das Gerätesystem ursalog 5000 umfaßt die speicherprogrammierbaren Steuereinrichtungen (SPS) ursalog 5010 und ursalog 5020. Die SPS ursalog 5000 werden für die Steuerung von Be- und Verarbeitungsmaschinen und Kinematiken der Roboter- und Handhabetechnik eingesetzt.

Die anwendungsrelevanten Parameter und Merkmale beider SPS sind in Tafel A1 dargestellt. Während die ursalog 5010 eine SPS der unteren Leistungsklasse darstellt, erlaubt die ursalog 5020 eine Anpassung an komplexe Steuerungsprobleme mittleren bis großen Umfangs.

Die SPS ursalog 5000 realisieren zyklisch die Funktionen

- Erfassung und Speicherung der Eingangssignale,
- Realisierung des zur Beschreibung des Steueralgorithmus erarbeiteten Anwenderprogramms,
- Ausgabe der Ausgabesignale.

Die Anwendungsgrenzen der SPS ursatron 5000 werden technisch bestimmt durch die

- maximale Zahl und Art der Eingangs- und Ausgangssignale,
- für die Aufnahme des Anwenderprogramms zur Verfügung stehenden Speicherkapazität,
- realisierbare minimale, durch die Verarbeitungszeit bedingte Reaktionszeit,
- Einsatzbedingungen.

Tafel A1. Parameter und Merkmale der SPS ursalog 5010

	SPS ursalog 5010	SPS ursalog 5020
Zentraleinheit	kompakte Verarbeitungseinheit 1 (VE 1)	modulare Verarbeitungseinheit 2 (VE2)
CPU	U880	MRS K1520 (U880)
Prozeßsignal-E/A-Anpassung	Baugruppensystem ursalog 4000	modulare Baugruppen des ursadat 5000 in Robustperipherie ursalog 5020
Anzahl binärer		
E-Signale	32	2048, 256 bei Robustperipherie davon 1 Byte je Zeit/Zählglied
A-Signale	16	
Merker	255	
Zeit/Zählglieder	16	
Analoge E/A	nein	ja
Programmsprache	PROLOG 1	PROLOG 2
Programmierung ausgehend von	Logikplan, Schaltgleichungen, PAP, PAG	
Programmiertechnik	Programmiergerät ursatron 5000	
Programminbetriebnahme	PROTES 5011	PROTES 5020, LOTES 5020
Inkrementelle Weg/Winkelgeber	nein	ja

Für die Programmerarbeitung, ausgehend von den üblichen Formulierungen des steuerungstechnischen Verhaltens der Einrichtung, dem Logikplan, den Schaltgleichungen, dem Programmablaufplan oder dem Programmablaufgraphen, wurde die Fachsprache PROLOG entwickelt, PROLOG 2 für die SPS ursalog 5020 und als Untermenge PROLOG 1 für die SPS ursalog 5010.

Die Programmerarbeitung für die SPS ursalog 5000 erfolgt auf dem Programmiergerät ursatron 5000 (PG ursatron 5000). Es realisiert die Umsetzung des in PROLOG formulierten Anwenderprogramms (AP) in das Objektprogramm (OP) in Maschinensprache. Dabei kommen die Compiler PROLOG 5010 und PROLOG 5020 zur Anwendung.

Das Programmiergerät ermöglicht in Kopplung mit den SPS auch die Inbetriebnahme und den Test des AP. Die Kopplung erfolgt über das serielle, sternförmige Interface IFSS gemäß TGL 29331. Dazu wurden die Softwarepakete PROTES 5011 und PROTES 5020 geschaffen. Sie gewährleisten in Verbindung mit dem PG ursatron 5000 eine vielfältige Beeinflussung der Abarbeitung des Anwenderprogramms und die schrittweise und segmentweise Kontrolle der Abarbeitung.

A2.1. Speicherprogrammierbare Steuereinrichtung ursalog 5010

Die SPS ursalog 5010 ist als robuste Steuereinrichtung für die Steuerung von Maschinen und Kinematiken der Handhabetechnik mit binären Signalgebern und binären Stellgliedern ausgelegt worden. Sie umfaßt die kompakte Verarbeitungseinheit 1 (VE 1) zur Realisierung der Steuerfunktionen sowie Baugruppen und Stromversorgungseinrichtungen des Baugruppensystems ursalog 4000 zur Realisierung der Prozeßsignalanpassung und der Stromversorgung.

Verarbeitungseinheit 1

Der Aufbau der VE 1 wird im Bild A2.1 in Blockdarstellung gezeigt.

Der interne funktionelle Ablauf sowie die Steuerfunktionen werden mit Hilfe der CPU (U880) und dem Zähler/Zeitgeber (U857) realisiert. Dazu erfolgt die Speicherung des Betriebssystems (LOGOS 5010) und von Unterprogrammen (1 KByte) sowie des OP (2 KByte) auf EPROM (U555). Zur Speicherung der binären Eingangsvariablen und Ausgangsvariablen sowie der bei der Lösung der Steuerfunktionen notwendigen operativen Variablen dient der RAM (U2, B2, 1 KByte).

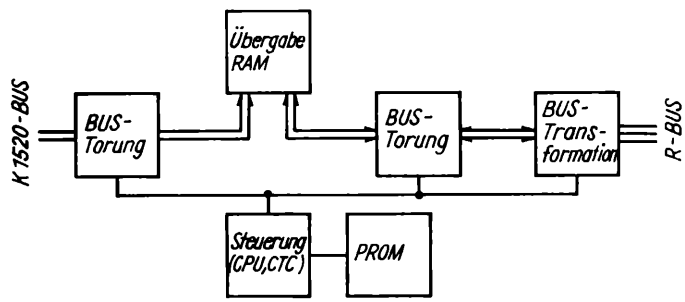


Bild A2.1
Verarbeitungseinheit 1 (VE 1)

Die binären Eingangssignale werden über die Eingangsfilter und Signalumschalter byteweise in den RAM übertragen. Die binären Ausgangssignale werden ebenfalls byteweise aus dem RAM gelesen und in die Ausgabespeicher übertragen.

Für die Impulsfrequenzmessung ($f \leqq 5$ kHz) oder die äquivalente Zeitmessung und die im Ergebnis der Meßvorgänge auszugebenden Entscheidungssignale wurden zusätzlich Eingänge und Ausgänge geschaffen. Signalanpassungsschaltungen realisieren die Filterung und Pegelanpassung sowie die zeitlich begrenzte Speicherung der Ausgabesignale.

Mit Hilfe der Schaltungsanordnung für die RESET-Erzeugung erfolgt die Initialisierung des Startzustands bei Netzeinschaltung sowie von Hand über eine Löschtaste.

Der fehlerfreie Lauf der VE 1 wird durch PROM-Test, RAM-Test, SV-Überwachung und Programmlaufzeitüberwachung umfangreich überwacht und gemeldet.

Die VE 1 wird übereinstimmend mit dem Baugruppensystem ursalog 4000 mit 24 V $\pm$ 25 % gespeist. Eine eigene Schaltungsanordnung gewährleistet die Erzeugung der erforderlichen internen, höherstabilisierten Speisespannungen.

Das Betriebssystem der VE 1 (LOGOS 5010) umfaßt die Programmteile für die Initialisierung der VE 1, die Übertragung der Eingangsvariablen in den RAM, der Ausgangsvariablen in die Ausgabespeicher sowie für die Steuerung der Abarbeitung des OP einschließlich des Einsatzes der Unterprogramme sowie des Ablaufs der Diagnoseprogramme und Testprogramme. Das OP für die Realisierung von Schaltgleichungen wird direkt durch den Mikroprozessor abgearbeitet; für die Realisierung der Zeit- und Zählfunktionen werden Unterprogramme aufgerufen.

In Ergänzung der VE 1 wurde für die Inbetriebnahme die Inbetriebnahmebaugruppe (VE 1I) entwickelt, die zur effektiven Inbetriebnahme des AP in Kopplung mit dem Programmiergerät ursatron 5000 dient. Sie ist anschlußkompatibel zur VE 1.

Für die Inbetriebnahme des Anwenderprogramms wurde die Testsoftware PROTES 5011 entwickelt. Sie gewährleistet in Verbindung mit dem Programmiergerät ursatron 5000 eine vielfältige Beeinflussung der Abarbeitung des Anwenderprogramms und die schrittweise und segmentweise Kontrolle der Abarbeitung.

Prozeßsignalanpassung

Die Prozeßsignalanpassung gewährleistet die Anpassung der Prozeßsignale an die Eingangssignale und Ausgangssignale der VE 1 und umfaßt die Pegelanpassung, galvanische Trennung, Verzögerung, Entprellung, Verstärkung, Speicherung und Anzeige von Signalen. Sie wird mit Hilfe von Baugruppen des Baugruppensystems ursalog 4000 realisiert, die auch eine der VE 1 vor- und nachgeschaltete Signalverknüpfung ermöglichen. Die Baugruppen besitzen eine ausgangsseitige Kurzschlußfestigkeit und eine hohe Stör- und Zerstörfestigkeit der Ein- und Ausgänge. Sie gewährleisten den Einsatz der SPS ursalog 5010 unter rauhen Industriebedingungen.

A2.2. Speicherprogrammierbare Steuereinrichtung ursalog 5020

Die SPS ursalog 5020 ist als modulare Steuereinrichtung für die komplexe Steuerung von Maschinen und Kinematiken der Robotertechnik ausgelegt worden. Sie gestattet die Lösung aller üblichen Steuerfunktionen binärer Steuerungen sowie arithmetischer Funktionen und die dazu erforderliche Erfassung und Ausgabe binärer, digitaler und analoger Signale. Die externe Parametrierung d. h. die Änderung der Sollwerte von Zeit- und Zählgliedern sowie die Änderung des Steueralgorithmus in Abhängigkeit von Prozeßsignalwerten und Bedieneingaben sind möglich. Die SPS ursalog 5020 umfaßt die modulare Verarbeitungseinheit (VE 2), Baugruppen zur Prozeßsignalerfassung und Ergänzungsbaugruppen sowie die zugehörigen Stromversorgungseinrichtungen.

Der Aufbau der SPS ursalog 5020 wird im Bild A2.2 in Blockdarstellung gezeigt.

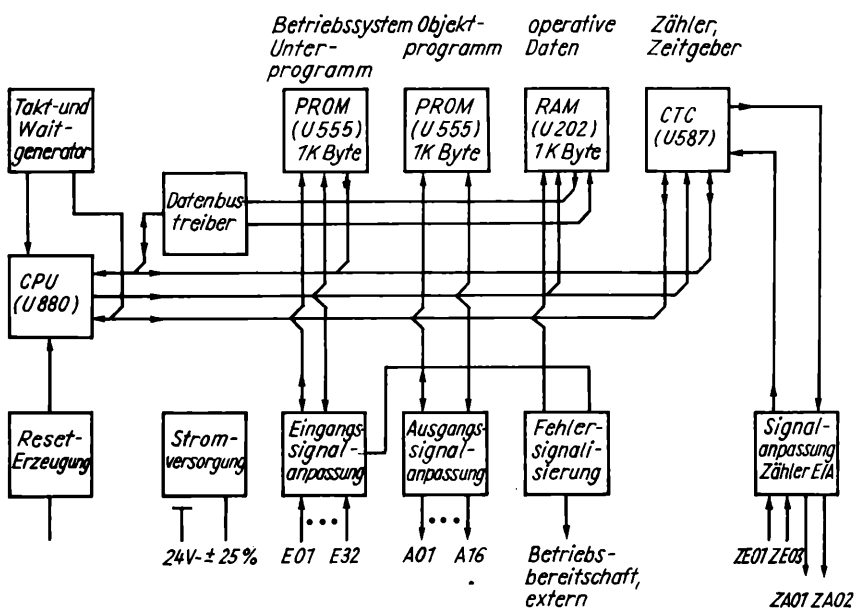


Bild A2.2. Aufbau der SPS ursalog 5020

Verarbeitungseinheit 2

Die VE 2 ist aus Baugruppen des Mikrorechnersystems K1520 und SPS-spezifischen Baugruppen aufgebaut. Sie erfüllen die Busrichtlinie TGL 37271 (K1520-Bus).

Als Baugruppen des MRS K1520 werden folgende Baugruppen in der VE 2 projektabhängig eingesetzt:

ZRE K2521.05 zentrale Recheneinheit mit LPU; 1 KByte statischer RAM, 3 KByte EPROM, 1 PIU, 1 LTL;
OPS K3520.05 4 KByte statischer RAM;
OPS K3521.05 4 KByte statischer RAM, CMOS, batteriegestützt;
OFS K3620.05 2 KByte statischer RAM, 6 KByte EPROM;
PFS K3620.05 16 KByte EPROM.

Die EPROM dienen zur Aufnahme des Betriebssystems und des OP, die RAM als Operativspeicher.

Die Initialisierung erfolgt bei Zuschalten der Netzspannung. Soll der erreichte Steuerzustand nach Ausfall der Netzspannung erhalten bleiben, so ist für die Speicherung der Eingabevariablen, Ausgabevariablen und Merkervariablen ein gestützter CMOS-RAM einzusetzen.

Als problemorientierte Baugruppe der VE 2 wurde die BG Kontrollmodul (KOMO, 3705) entwickelt, der den Anschluß eines Tastatur-Anzeige-Vorsatzes (TAV, 3726) erlaubt.

Im einzelnen werden folgende Überwachungsfunktionen realisiert:

- Überwachung der Abarbeitung der Programme des Betriebssystems, des Anwenderprogramms sowie von Dienstprogrammen durch Analyse der vom Betriebssystem erzeugten Fehlerbytes;
- Überwachung der von den SV-Einrichtungen abgegebenen Meldesignale und Bildung von Steuersignalen für die Steuerung des Prozessors der ZRE zum Autostart oder Start einer Netzausfallroutine;
- Festlegen der Betriebsart der SPS ursalog 5020.

Es sind dies

LAUF Abarbeitung des Anwenderprogramms;
TEST Aktivierung der Schnittstelle IFSS bei Anschluß des Programmiergeräts (durch PROTES beeinflußter Lauf des AP);
HAND Arbeit mit LOTES;
RETT Initialisierung des Laufs eines Anwenderprogramms (wird gewählt nach erkanntem Fehler im Prozeß).

Der KOMO ist ein problemorientierter Einkartenmikrorechner (CPU, 1 KByte PROM, CTC, PTO, 1 KByte RAM). Der TAV kann direkt auf die Baugruppe KOMO gesteckt werden oder über ein 1,5-m-Adapterkabel angeschlossen werden. Es erlaubt den Anschluß eines Kassettenmagnetbandgeräts als Massenspeicher.

Der TAV trägt den

- Betriebsartenschalter;
- acht 7-Segment-Anzeigeelemente;
- zwei Luminiszenzdioden (1 × für Spannungskontrolle, 1 × für Pegelkontrolle bei Kassettenmagnetbandeingabe);
- 21stellige Textatur (hexadezimal + Sonderzeichen).

Die Überwachung der Programme erfolgt durch Überwachung des zeitlichen Abstands der von der ZRE generierten Fehlerbyte ($\leq$26 ms); erkannte Fehler werden auf dem TAV angezeigt.

Ein-/Ausgabe-Baugruppen

Die Ein-/Ausgabe-Baugruppen (E/A-BG) gewährleisten die Kopplung der SPS mit dem Prozeß.

Es bestehen zwei gerätetechnische Lösungen: zum einen die Verwendung von E/A-BG, die direkt am K1520-Bus arbeiten und denen ggf. Baugruppen des BG-Systems ursalog 4000 vor- oder nachgeschaltet werden, und zum anderen die Verwendung der E/A-BG der Robustperipherie ursalog 5020.

E/A-Baugruppen mit K 1520-Bus

Die Tafel A2 enthält eine Auswahl der in SPS ursalog 5020 häufig angewendeten E/A-BG mit K1520-Bus. Sie erlauben die Eingabe und Ausgabe analoger und digitaler Signale.

Für den Anschluß inkrementeller Geber wurde eine Mikrorechnerbaugruppe (IIM) entwickelt, die von der ZRE wie eine E/A-BG behandelt wird. Sie übernimmt Sollwerte der Wege oder Winkel und erfaßt die von den Gebern abgegebenen Impulsfolgen. Bei Erreichen der vorgegebenen Sollwerte wird anwendungsabhängig ein Interruptsignal gegeben oder eine vereinbarte Marke in den RAM abgelegt, die von der ZRE gelesen werden kann [8.30].

Tafel A2. E/A-Baugruppen mit K1520-Bus

Typ	Kurzcharakteristik
DES 2340	Digitaleingabe statisch, 16 Bit, optoelektronische Trennung, Pegel: 5, 12, 24, (60) V Gs
DEM 2341	Digitaleingabe multiplex, Matrixabfrage von 128 Bit Kontaktsignalen, optoelektronische Trennung
DES-KT 2344	Digitaleingabe statisch mit Systempegel ursalog 4000 32 Bit, geeignet zur Eingabe von Signalen, die von der BG ursalog 4000 abgegeben werden
DA-R 2331	Digitalausgabe mit Relais RGK 20/1, 24 bit, max. 60 V Gs/Ws/0,5 A/10 W
DAS-KT 2334	Digitalausgabe statisch mit Systempegel ursalog 4000 32 Bit, geeignet zur Ausgabe von Signalen an BG ursalog 4000
DA-O 2335	Digitalausgabe mit Optokoppler, 16 Bit, optoelektronische Trennung, Transistorausgang max. 60 V Gs/0,1 A/3 W
DA-T 2336	Digitalausgabe mit Transistor, 32 Bit, max. 60 V Gs/0,12 A/7,2 W
SE-AS 2351	Service-Einheit-Anschlußsteuerung
AE-G 2310	Analogsignaleingabe, 8 Kanäle multiplex, Analogsignal: 1 V, Digitalisierung: 12 Bit dual, Umsetzungszeit: 25 µs, Fehlerklasse: 0,25; 0,4; 0,6 geberabhängig. Weitere Baugruppen realisieren: Signalverstärkung, Erhöhung der Kanalzahl, galvanische Trennung, Anpassung passiver Geber u.a.
AA-1K2301	Analogsignalausgabe 1 Kanal, Digitalsignal: 10 Bit dual, Analogsignal: 10 V; 5 mA; 1–5 mA; 20 mA; 4–20 mA; Fehlerklasse: 0,1; Umsetzungszeit: 200 µs
IIM 3760	Inkrementsignalfolgeeingabe für 3 inkrementelle Geber IGR oder IDL 500 des VEB Carl Zeiss JENA.

E/A-Baugruppen der Robustperipherie

Die Robustperipherie erlaubt den Aufbau einer selbständig arbeitenden E/A-Signal-Anpassungseinheit. Sie besteht aus einer Mikrorechner-BG, der Robustbussteuereinheit (R-Bus-S, 3725), die am K1520-Bus der VE 2 arbeitet und den Robustbus (R-Bus) ge-

neriert. Die E/A-BG der E/A-Signal-Anpassungseinheit können bis zu 200 m von der R-Bus-S abgesetzt angeordnet werden. Der R-Bus und die Prozeßsignaleingänge und -ausgänge besitzen eine hohe Stör- und Zerstörfestigkeit. Der R-Bus und die Prozeß signalausgänge sind kurzschlußfest.

Tafel A3. E/A-Baugruppen der Robustperipherie ursalog

Typ	Kurzcharakteristik
R-BUS-S 3725	Robust-BUS-Steuerung, generiert aus dem K1520-BUS den Robust-BUS am Frontstecker, Diagnose-Funktionen
R-BUS-A 3721	Robust-BUS-Anschluß, führt den Robust-BUS vom Frontstecker auf die Rückverdrahtung
R-BUS-TR 3720	Robust-BUS-Treiber, wie R-BUS-A, jedoch mit Signalregenerierung
R-DES-KT 3730	Digitaleingabe statisch mit Systempegel ursalog 4000, 32 Bit
R-DES-K 3731	Kontakteingabe statisch, 24 V, Kontaktbelastung 10 mA, 32 Bit
R-DES-I 3732	Eingabe von Signalen von Initiatoren des VEB MGB, 24 Bit
R-DAS-3A 3741	Schaltverstärker 24 V ± 25%/3A, Gs, 6 Bit
R-DAS-TC 3742	Schaltverstärker mit Triac 220 V Ws + 10% − 15%/3 A, 4 Bit, Schmelzsicherung je Ausgang mit Sicherungsanzeige
R-DAS-RI 3743	Relais-Ausgabe über GBR 10, 8 Bit (Wechsler), max. 30 V, 0,1–3 A
R-DAS-RU 3744	wie R-DES-RI, jedoch 220 V Ws/150 VA/cos φ = 1, 220 V Ws/100 VS/ cos φ = 0,4
R-DAS-0,25 3745	Schaltverstärker 24 V ± 25%/0,25 A, 14 Bit

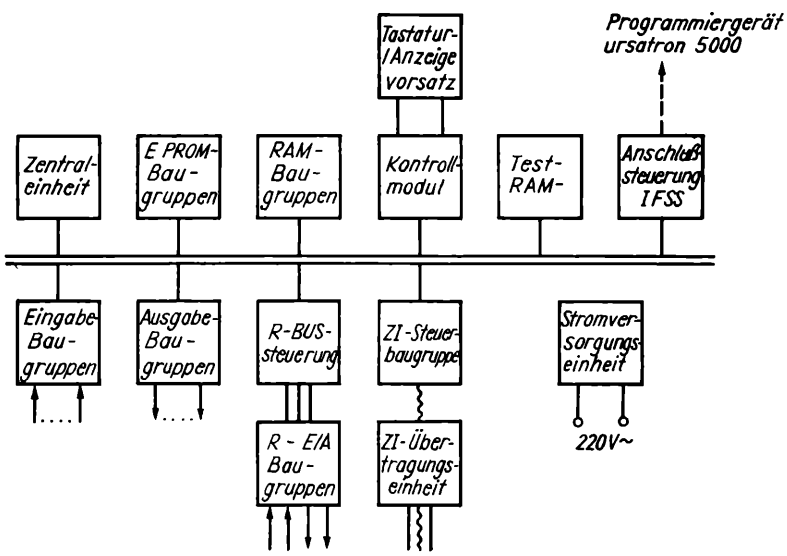


Bild A2.3. Kopplung des K1520-Bus mit dem R-Bus

Tafel A3 enthält die Baugruppen der Robustperipherie. Die BG der Robustperipherie haben für alle E/A-Signale eine Statusanzeige und werden aus einer unstabilisierten Stromversorgung gespeist. Die Robustperipherie erlaubt eine umfangreiche Eigendiagnose und die Erkennung von Kurzschlüssen der Ausgabestromkreise.

Die R-Bus-S gewährleistet eine von der VE 2 unabhängige Arbeitsweise. Das Erfassungs-, Ausgabe- und Diagnoseprogramm ist PROM-resident. Die E-Signal-Zustände werden zyklisch erfaßt und in einem RAM-Bereich abgelegt und die A-Signale zyklisch aus einem RAM-Bereich gelesen und ausgegeben. Die Ergebnisse der Eigendiagnose werden in einem weiteren RAM-Bereich abgelegt. Diese RAM-Bereiche werden von der VE 2 gelesen bzw. beschrieben.

Das Interface R-Bus wird mit Hilfe spezieller Bustreiber (D 412) realisiert. Die Datenübertragung erfolgt getrennt über jeweils acht Leitungen für die Dateneingabe und -ausgabe. Die Adresse wird über 16 Leitungen (Blockadresse, Moduladresse, Subadresse) übertragen; es werden vier Steuerleitungen angewendet. Die Moduladresse wird durch die gedruckte R-Bus-Verdrahtung festgelegt; eine Adreßfestlegung auf der BG ist nicht erforderlich.

Bild A2.3 zeigt in Blockdarstellung die Kopplung des K1520-Bus und des R-Bus.

A2.3. Programmsystem der speicherprogrammierbaren Steuereinrichtung ursalog 5020

Um den Anforderungen mittelgroßer bis großer Steuerungsprobleme und auch der Steuerung von Industrierobotern gerecht zu werden, ist die SPS ursalog 5020 serienmäßig mit einem Echtzeitsteuerprogrammsystem für Mehrprogrammbetrieb (Multitask-Echtzeit-Betriebssystem) ausgestattet. Das Programmsystem PVE2 enthält alle Programme, die in einer SPS ursalog 5020 laufen sollen. Das sind die anwendungsunabhängigen Echtzeitbetriebsprogramme EIEX 1521 und LOGOS 5020 einschließlich spezieller Dienstprogramme, das Testsystem LOTES 5020 sowie die Anwendersteuerprogramme LOGAN 5020.

EIEX 1521 ist eine speziell generierte Version des internspeicherorientierten Echtzeitsteuerprogrammsystems EIEX K1521 und fungiert als Rahmensteuerprogrammsystem mit den wesentlichen Funktionskomponenten:

- Anlauforganisation
- Interruptorganisation
- Vorrangorganisation
- Zeitorganisation
- Ruforganisation
- Ein-/Ausgabe-Organisation der Prozeßperipherie.

Die Programme LOGAN 5020, LOTES 5020 sowie einige Programme von LOGOS 5020 laufen unter der Regie von EIEX 1521 im Sinne von Anwendertasks. Eine Task ist ein lauffähiges Objektprogramm mit eigenem Stackbereich und definiertem Einsprungspunkt. Die Tasks sind gegeneinander priorisiert und können zyklisch oder durch Interrupt oder von anderen Tasks gestartet werden. Die Priorität einer Task, ausgedrückt durch eine Zahl <128, ist gleichzeitig ihr Identifikator. Die höchste Priorität besitzt die Task 1.

LOGAN 5020 ist der Sammelbegriff für die in der SPS laufenden Anwendersteuerprogramme (AP), die gewöhnlich vom Anwender in der Fachsprache PROLOG 2 notiert und mit dem Programmiergerät ursatron 5000 übersetzt werden. Der Anwender kann mit der Standardversion des PVE2 bis zu sechs Programme (Tasks) mit unterschiedlicher Priorität und ggf. Zykluszeit ins Programmsystem aufnehmen. Die Zuordnung von Priorität und Zykluszeit erfolgt in der Programmnotation und muß unter Berücksichtigung der bereits im System enthaltenen Tasks vorgenommen werden. Darüber hinaus können

Tafel A4. Task-Übersicht

TASK-Nr.	TASK-Name	System-name	Zyklus-zeit	Laufzeit	
1		LOGOS			
2	FAUS	LOGOS			Fehlerausgabe Systemnachrichten EIEX/LOGOS
3[1])	FEBE	LOGAN			Fehlerbehandlung kann Anwender schreiben (APRO/E 3)
4[1])	INIT	LOGAN			Initialisierung kann Anwender schreiben (APRO/E 4)
5	ANME	LOGOS	1 s		Programm-Anmeldung und BAS-Abfrage auf „RETT“
6[1])	RETT	LOGAN		max. 3,5 ms	Rettungsprogramm bei BAS-„RETT“ kann Anwender schreiben (APRO/E 6)
7	ZYKL	LOGOS			Zyklusüberwachung/-anzeige
8[1])	IMES	LOGOS	20 ms		Erfassungsprogramm Inkrementelles Meßsystem
9	BLS1	LOGOS			Bedienung Langsame Serielle Schnittstelle 1 Slave)
10[1])	BLS2	LOGOS			Bedienung Langsame Serielle Schnittstelle 2 (Slave)
11	TESH	LOTES			Testsystem – HAND
12[1])		LOGAN			APRO 12 oder APRE 12
13[1])	EMTA	LOGOS			DUE-Empfangstask
		LOGAN			APRO 13 oder APRE 13
14[1])	ROBU	LOGOS	50 ms		Robustperipherie-Erfassungsprogramm
		LOGAN			APRO 14 oder APRE 14
15[1])	SGMT	LOGOS	50 ms		SGM-Task
		LOGAN			APRO 15 oder APRE 15
16[1])		LOGAN			APRO 16 oder APRE 16
17[1])		LOGAN			APRO 17 oder APRE 17
⋮					
n – 2	TESL	LOTES	100 ms		Testsystem – LAUF
n – 1	TROM	LOGOS	10 s	15 ms/KByte	Test-/Prüfprogramm PROM
n	TRAM	LOGOS	10 s	165 ms/KByte	Test-/Prüfprogramm RAM

[1]) Die Übernahme bzw. Auswahl der Task für das Programmsystem ist optional und wird bei der Generierung des Betriebssystems festgelegt.

wahlweise je ein Programm zur anwendungsspezifischen Netzausfallrettung, Fehlerbehandlung, Initialisierung und zum Übergang in einen sicheren Prozeßzustand (ausgelöst durch BAS = RETT) ins PVE2 aufgenommen werden.

Die Programme des Testsystems LOTES 5020 realisieren Bedien-, Test- und Inbetriebnahmefunktionen, die mittels TAV des Kontrollmoduls oder vom angeschlossenen Programmiergerät ausgelöst werden.

Der Anwender hat die Möglichkeit, beim Hersteller der SPS aus einer Menge von Standardversionen des PVE2 die für ihn optimale Version auszuwählen. Er kann sich aber auch eine auf seinen Anwendungsfall speziell zugeschnittene Version generieren lassen. Einige der wesentlichen Angaben für die Generierung, die auch entscheidend den Umfang der Betriebssystemkomponenten bestimmen, sind

- Benutzung der zweiten langsamen seriellen Schnittstelle,
- Anzahl der aufzunehmenden Anwendersteuerprogramme,
- Typen und Anzahl von Geräten der Prozeßperipherie,
- Typen von Sprachmodulen entsprechend dem verwendeten Sprachumfang von PROLOG 2,
- Typen von Modulen des Testsystems LOTES 5020.

In Tafel A4 sind die Tasks eines PVE2 für ein Anwendungsbeispiel der SPS ursalog 5020 zusammengestellt.

A2.4. Fachsprache PROLOG 2

Zur problemorientierten Notation von Steuerprogrammen wurde die Fachsprache PROLOG 2 entwickelt [8.34]. Sie ist aufwärtskompatibel zur Fachsprache PN 5000 für die neue Anlagengeneration NAG [8.35]. Bei der Konzipierung wurde von den Notwendigkeiten der Anwendung für die SPS ursalog 5020 ausgegangen. Trotzdem weisen die in PROLOG 2 notierten Anwendersteuerprogramme (Quellprogramme) kaum maschinenspezifische Besonderheiten auf. Sie können in Verbindung mit einem maschinenspezifischen Compiler als Quellprogramme für beliebige SPS dienen – vorausgesetzt, die SPS enthält ein hierfür vorbereitetes Echtzeitbetriebssystem und die notwendigen Softwaremodule. PROLOG 1, eine Untermenge von PROLOG 2, ist die Fachsprache zur Notation von AP für eine SPS ursalog 5010 [8.33].

In PROLOG 2 wird das Steuerprogramm ausgehend von einer strukturbezogenen Darstellung des Steueralgorithmus als eine Folge von Vereinbarungen, Funktions- und Bausteinanweisungen notiert. Als strukturbezogene Darstellung (Algorithmusdarstellung mit den Strukturvariablen) eignen sich Strukturgleichungssysteme bzw. Schaltgleichungen, Signalflußbild, Programmablaufplan bzw. -graph. Bei Beachtung von Umformungsregeln kann auch von einem Kontaktnetzwerk ausgegangen werden. Für anspruchsvolle Probleme besteht auch die Möglichkeit, mittels sog. Betriebssystemaufrufe den zeitlichen und priorisierten Lauf von Anwenderprogrammen zu modifizieren.

Programmstruktur

Ein in PROLOG 2 notiertes Programm besteht aus Vereinbarungen und Anweisungsfolgen. In den Vereinbarungen werden die Art der Prozeßsignaleingabe und -ausgabe und Initialwerte für Variablen festgelegt.

Eine Anweisungsfolge ist die äquivalente Abbildung eines Steueralgorithmus (oder eines Teiles davon), bestehend aus einer Folge von Anweisungen für die Ausführung von informationsverarbeitenden und programmorganisatorischen Funktionen. Die informa-

tionsverarbeitenden Funktionen werden in der SPS durch die Abarbeitung von parametrierbaren Befehlsfolgen (Unterprogramme im Betriebssystem oder direkt Befehlssequenzen im Anwendermaschinenprogramm) realisiert. Diese Befehlsfolgen werden unabhängig von der Art ihrer Ausführung Programmodule genannt. Dementsprechend stellen die im Quellprogramm zu notierenden Anweisungen Modulaufrufe dar. Der größte Teil der programmorganisatorischen Funktionen wird in Komponenten des SPS-Betriebssystems (EIEX 1521) realisiert. Die Anweisungen für diese Funktionen werden Betriebssystemaufrufe genannt.

Alle Anweisungen bestehen aus einem Operationsteil und einem Operandenteil. Der Operationsteil kennzeichnet die auszuführende Operation. Der Operandenteil enthält die für die Ausführung der Operation notwendigen Parameter. Einer Anweisung kann optimal eine Marke in Form einer 3stelligen Dezimalzahl vorangestellt werden. Diese Marke dient für Sprungoperationen als Kennzeichen (Parameter im Modulaufruf) der Ansprungstelle.

Zur verständlichen Programmdokumentation sind in allen Programmzeilen Kommentare zugelassen.

Eine oder auch mehrere Anweisungsfolgen können als Unterprogramme deklariert werden. Unterprogramme können von beliebiger Stelle des Programms oder auch anderer Programme zur Abarbeitung aufgerufen werden.

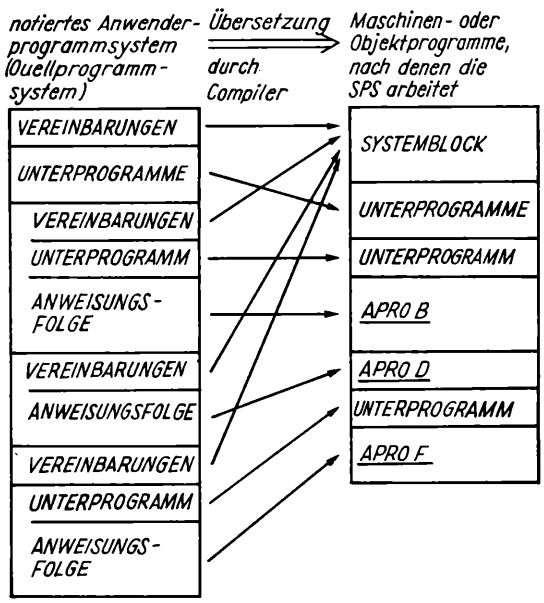


Bild A2.4
Zusammenhang zwischen Quell- und Objektprogrammsystem

In eine SPS ursalog 5020 können mehrere Programme der beschriebenen Struktur nach ihrer Übersetzung eingebunden und zur Abarbeitung gebracht werden. Der Gesamtmenge von Programmen in ihrer Quellnotation können auch noch Vereinbarungen der eingangs genannten Typen vorangestellt werden. Alle Vereinbarungen, die allen Programmen vorangestellten und die am Anfang eines Programms notierten, sind gleichrangig und gelten für alle Programme; sie haben globalen Charakter. Bei der Übersetzung der Programme (einschließlich der vorangestellten Vereinbarungen) auf dem Programmiergerät ursatron 5000 wird ein gemeinsamer Systemblock erstellt. Der Systemblock ist ein formatierter Datenbereich, der die globalen Größen (z. B. Adreßvereinbarungen für

die Variablenfelder) und Steuerinformationen (z.B. Programm- und Unterprogrammverzeichnis, Prozeßperipherie-Tab.) enthält. Er wird der SPS mit den Objektprogrammen als Bestandteil des Anwenderprogrammsystems LOGAN 5020 übergeben. Im Systemanlauf der SPS beziehen die Programme zur Systeminitialisierung (Komponenten von LOGOS 5020) wesentliche Informationen über die einzubindenden Anwenderprogramme aus dem Systemblock.

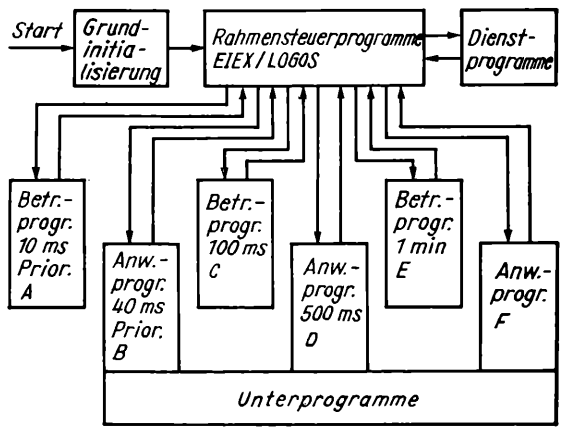


Bild A2.5
Beispiel für die Struktur eines Programmsystems

Bild A2.4 verdeutlicht den Zusammenhang zwischen der Quellnotation und der Maschinendarstellung. Mit diesem AP-System ist im Bild A2.5 in vereinfachter Weise die Struktur eines SPS-Programmsystems dargestellt. Die Prioritäten sind nur symbolisch durch Buchstaben markiert; die Tasks des Betriebssystems werden nur durch drei Tasks repräsentiert. In dem Block Dienstprogramme sollen auch alle Sprachmodule und Bedienprogramme für Geräte der Prozeßperipherie enthalten sein.

Konstanten und Bezeichner für Variable

Konstanten werden zur Parametrierung von Vereinbarungen und Anweisungen verwendet. Sie können in unterschiedlicher Darstellung vorgegeben werden:

Dualkonstanten, z.B. ØB, 1B, 1Ø11Ø1B, 1ØØØ11Ø11B
Oktalkonstanten, z.B. 55Q, 433Q
Dezimalkonstanten, z.B. 45, 3Ø7, −42, −215ØØ
Hexadezimalkonstanten, z.B. 2DH, 11BH, ØA22FH.

Negative Dezimalkonstanten werden nur in Arithmetikmodulen verwendet.

Bezeichner für Variable dienen der Adressierung von Bit-, Byte- und Doppelbytevariablen eines Anwendersteuerprogrammsystems. Variable werden als Operanden in den Vereinbarungen und Anweisungen verwendet und sind globale Größen, d.h., sie bezeichnen in verschiedenen Programmen jeweils ein und dasselbe Datenwort. Folgende Variablenbezeichner sind verfügbar:

Bitvariable Ø,1

Ei Eingangsvariable; mittels Prozeßperipherie erfaßtes Binärsignal
Aj Ausgangsvariable; mittels Prozeßperipherie auszugebendes Binärsignal
Mk Merker (Bit); intern erzeugtes Zwischensignal
ETl Eingang für Zeitmodul *l*

ATl Ausgang von Zeitmodul *l*
RTl Rücksetzeingang für Zeitmodul *l*
EZm Eingang für Zählmodul *m*
AZm Ausgang von Zählmodul *m*
RZm Rücksetzeingang für Zählmodul *m*
UZm Umschalteingang für Zählmodul *m*
AC Akkumulator für Bitlogikmodule (temporärer Merker)
SM binärer Sondermerker.

Beispiele

E1 Bit 1 von EBØ
E371 Bit 3 von EB46
A54 Bit 6 von AB6
ET7 Eingang für Zeitmodul 7
AZ12 Ausgang von Zählmodul 12.

Da die Variablen Ei, Aj und Mk gleichzeitig auch Bits von Bytevariablen EBi*, ABj* und MBk* sind, ist unbedingt der Zusammenhang zwischen den Indizes zu berücksichtigen:

$$i^* = \text{integer}\,[\tfrac{i}{8}] \quad \text{usw.}$$

Bytevariable 0 bis 255

EBi* Eingangsvariable (jeweils acht Bitvariable *E* enthaltend)
ABj* Ausgangsvariable (jeweils acht Bitvariable *A* enthaltend)
MBk* Merker (Byte; jeweils acht Bitvariable *M* enthaltend) für intern erzeugte Zwischengrößen
FZh Fehlerzelle
ITl Istwert des Zeitmoduls *l*
DTl Sollwert für Zeitmodul *l* (Verzögerungs- oder Impulsdauer)
PTl zusätzlicher Sollwert für Zeitmodul *l* vom Typ VIBi (Impulspause)

EDr.Ø unteres
EDr.1 oberes } Byte von Doppelbyteeingangsvariable EDr

ADs.Ø unteres
ADs.1 oberes } Byte von Doppelbyteausgangsvariable ADs

MDt.Ø unteres
MDt.1 oberes } Byte vom Doppelbytemerker MDt

IZm.Ø unteres
IZm.1 oberes } Byte vom Istwert (zwei Byte) des Zählmoduls *m*

GZm.Ø unteres
GZm.1 oberes } Byte vom Grenzwert (zwei Byte Sollwert) des Zählmoduls *m*

Beispiele

EB46 Eingangsvariable mit E368 bis E375
AB6 Ausgangsvariable mit A48 bis A55
DT7 Sollwert für Zeitmodul 7

ED14.Ø unteres } Byte der Doppelbyteeingangsvariablen ED14
ED14.1 oberes }
GZ12.Ø unteres Byte vom Grenzwert des Zählmoduls 12

Doppelbytevariable Ø bis {65535}

EDr Eingangsvariable
ADs Ausgangsvariable
MDt Merker (zwei Byte); für intern erzeugte Zwischengrößen
IZm Istwert (zwei Byte) des Zählmoduls *m*
GZm Grenzwert (zwei Byte Sollwert) für Zählmodul *m*

Beispiele

ED14 Doppelbyteeingangsvariable, bestehend aus den Bytevariablen ED14.Ø und ED14.1
GZ12 Grenzwert für Zählmodul 12, bestehend aus den Bytes GZ12.Ø und GZ12.1.

Vereinbarungen

Die folgenden fünf Typen von Vereinbarungen können an beliebiger Stelle des Programmsystems stehen, in der Regel jedoch allen Programmen vorangestellt oder jeweils im Programm direkt nach der Programmeröffnung. Diese Vereinbarungen finden durch den Compiler nur Berücksichtigung bei der Erstellung des Systemblocks.

PEAV Prozeß-Ein-/Ausgabe-Vereinbarung
zur anwendungsspezifischen Zuordnung von Prozeßsignalen zu Variablen eines SPS-internen Abbildspeichers über Eingabegeräte und der umgekehrten Zuordnung über Ausgabegeräte
DEF1 Initialisierungsvereinbarung für Bytevariablen
z.B. DEF1 AB11, 37H, 123Q, 2Ø9
;AB11:=37H, AB12:=123Q, AB13:=2Ø9
DEF2 Initialisierungsvereinbarung für Doppelbytevariablen,
z.B. DEF2 MD23, 49H, ØB43FH
;MD23:=49H, MD24:=0B43FH, bzw. MD23.Ø:=
;49H, MD23.1:=Ø, MD24.Ø:=3FH, MD24.1:=ØB4H
DEF2 AB11, 5337H, 2Ø9
;AB11:=37H, AB12:=53H=123Q, AB13:=2Ø9=
;ØD1H, AB14:=Ø
APRE Vereinbarung zur Einbindung eines prioritätsgesteuerten externen Programms (Maschinenkodeprogramm) ins Programmsystem
IPRE Vereinbarung zur Einbindung eines interruptgesteuerten externen Programms (Maschinenkodeprogramm) ins Programmsystem.

Mit den nachfolgenden Vereinbarungen werden PROLOG-2-Programme eröffnet. Die Angaben in den Vereinbarungen bezeichnen das jeweilige Programm und legen Modalitäten der Abarbeitung fest:

APRO Vereinbarung eines prioritätsgesteuerten Programms
IPRO Vereinbarung eines interruptgesteuerten Programms
UPRO Vereinbarung eines Unterprogramms.

Anweisungen

Anweisungen sind Aufrufe von Programmodulen und Betriebssystemrufe. Programmmodule sind entweder vom Compiler ins Objektprogramm generierte Prozeduren im Sinne von Makros (u.a. alle Logik- und Transportmodule) oder Unterprogramme als fester Bestandteil von LOGOS 5020. Die Anweisungen dazu bestehen jeweils aus dem Modulbezeichner und einer Folge von Operanden.

Prozeß-Signalein-/-ausgabe-Module

PRIN Prozeßsignaleingabe in Regie von EIEX
PROU Prozeßsignalausgabe in Regie von EIEX
PRES schnelle Prozeßsignaleingabe
PRAS schnelle Prozeßsignalausgabe.

Transportmodule

UMBI Wertzuweisung an Bitvariable; bedingt oder unbedingt,

z.B. UMBI M27,/A13, AT3

;$\overline{A13}$:=M27 bzw. A13:=$\overline{M27}$ für AT3=1

;A13 unverändert für AT3=Ø

UMD1 Wertzuweisung an Bytevariable; bedingt oder unbedingt,

z.B. UMD1 1Ø1Ø1Ø1ØB,AB9,E117

;AB9:=ØAAH, wenn E117=1, ansonsten unverändert

UMD1 MB27,MB37; MB37:=MB27

UMUMD2 Wertzuweisung an Doppelbytevariable;
bedingt oder unbedingt.

Bitlogik-, Bytelogik-, Zeit- und Zählmodule sind in Tafel A5 dargestellt. Ihre Funktion ist äquivalent der bekannter Hardwarebausteine. In der Tabelle entspricht die symbolische Darstellung dem jeweiligen Beispiel.

Vergleichsmodule

VGL1 Vergleich zweier Byteoperanden auf Gleichheit
VGL2 Vergleich zweier Doppelbyteoperanden auf Gleichheit
VGR1 Vergleich zweier Byteoperanden auf „größer“,

z.B. VGR1 EB16,1ØØ,/M13

;$\overline{M13}$:=1 für EB16 > 1ØØ

;$\overline{M13}$:=Ø für EB16 ≦ 1ØØ

VGR2 Vergleich zweier Doppelbyteoperanden auf „größer“,

z.B. VGR2 ED13,MD29,A96

;A96:=1 für ED13 > MD29

;A96:=Ø für ED13 ≦ MD29

Tafel A5. Programmodule

Modul	Symbol	Beispiel/Erläuterung
BITLOGIKMODULE		
UNDB	M1 E15 E7 ET2Ø & A13	UNDB M1,E15,/E7,ET2Ø,A13 4-Eingangs-UND-Gatter, wobei ein Eingang negiert
ODRB	M3 AZ6 1 M1Ø	ODRB M3,AZ6,/M1Ø 2-Eingangs-NOR-Gatter
XORB	E12 AT9 A17 M2 RZ4	XORB E12,AT9/A17,RZ4 Addition modulo 2 (serielle Antivalenz, Paritätsdetektor) mit drei Eingängen
BOOL	x_i f(x) y $\bar{y}$	BOOL $y = f(x_0, \ldots, x_i, \ldots, x_{n-1})$ Ermittlung des Wertes einer Schaltfunktion (Boolescher Ausdruck für mehrstufige UND-ODER-NICHT-Logik)
BYTELOGIKMODULE		
UNDD	ED5.1 8 3Ø7Q 8 8c AB12 8	UNDD ED5.1,3Ø7Q,AB12 8fach UND-Verknüpfung (bitweise je Byte)
ODRD	EB37 8 MB4 8 1 MB4 8	ORDD EB37,MB4,MB4 8fach ODER-Verknüpfung (bitweise je Byte)
XORD	MB4 8 ØC7H 8 M2 AD3.Ø 8	XORD MB4,ØC7H,AD3.Ø 8fach Addition modulo 2
ZEITMODULE		
TAN1	DT/Ø ET13 RT13 — TAN1 Nr.13 AT13	TAN1 13,8Ø,CS Zeitglied Nr.13 mit 0,8 s Anzugsverzögerungszeit
TAB1	Ø/DT ET4 RT4 — TAB1 Nr.4 AT4	TAB1 4,1ØØ,DS Zeitglied Nr. 4 mit 10 s Abfallverzögerungszeit
IPG1	DT ET21 RT21 S IPG1 Nr.21 AT21	IPG1 21,12Ø,S Zeitglied Nr.21 (Impulsgeber – MMV) mit 2 min Impulsdauer
VIB1	DT/PT ET17 RT17 6 VIB1 Nr.17 AT17	VIB1 17,1Ø,9Ø,CS Zeitglied Nr.17 (steuerbarer astabiler Multivibrator – AMV) für 1 Hz mit Tastverhältnis 1:9
ZÄHLMODULE		
ZVO1	EZ9 RZ9 ST2 ZVO1 Nr.9 AZ9	ZVO1 9,175 Zählglied Nr.9 (Vorwärtszähler) mit Zählgrenzwert GZ9=175 (max. 1 Byte)
ZVO2	EZ5 RZ5 CT2 ZVO2 Nr.5 AZ5	ZVO2 5,4711 Zählglied Nr.5 (Vorwärtszähler) mit Zählgrenzwert GZ5=4711 (max. 2 Byte)
ZVR2	EZ17 UZ17 RZ17 CT2 ZVR2 Nr.17 AZ17	ZVR2 17,288 Zählglied Nr.17 (Vor-/Rückwärtszähler) mit Zählgrenzwert GZ17=288 (max. 2 Byte)

Programmorganisationsmodule

SPRG Sprung zu einer Zielmarke; bedingt oder unbedingt,

z.B. SPRG 21 ;unbedingter Sprung zur Marke 21

SPRG 22,E53 ;Sprung zur Marke 22, wenn E53/1

SPUP Unterprogrammaufruf; bedingt oder unbedingt,

z.B. SPUP OTTO ;unbedinger Aufruf des Unterprogramms

;OTTO (UP beginnt mit der Zeile

;„UPRO OTTO")

UEND Endemarkierung für Unterprogramm
PEND Endemarkierung für prioritätsgesteuertes Programm
IEND Endemarkierung für interruptgesteuertes Programm.

Arithmetikmodule

Sie verarbeiten vorzeichenbehaftete Rechtskommadualzahlen (Integer-Arithmetik):

ADD1 Addition von Doppelbyteoperanden,

z.B. ADD1 EB14,115,MB15 ;⟨MB15⟩ := ⟨EB14⟩ +115

ADD2 Addition von Doppelbyteoperanden
SUB1 Subtraktion von Byteoperanden,

z.B.: SUB1 MB11, EB9, MB12 ;⟨MB12⟩ := ⟨MB11⟩ −⟨EB9⟩

SUB2 Subtraktion von Doppelbyteoperanden
MUI1 Multiplikation von Byteoperanden
MUI2 Multiplikation von Doppelbyteoperanden
DVI1 Division von Byteoperanden
DVI2 Division von Doppelbyteoperanden
INK1 Inkrementieren einer Bytevariablen
INK2 Inkrementieren einer Doppelbytevariablen
DEK1 Dekrementieren einer Bytevariablen
DEK2 Dekrementieren einer Doppelbytevariablen.

Konvertierungsmodule

DUD1 Dualzahl in 3stellige BCD-Zahl
DUD2 Dualzahl in 5stellige BCD-Zahl

z.B. DUD2 MD9,AB8 ;für z.B. MD9=7C8ØH = 31872

;ergibt die Konvertierung:

; AB8:/Ø111ØØ1ØB ≙ 72

; AB9:=ØØØ11ØØØB ≙ 18

;AB1Ø:=ØØØØØØ11B ≙ Ø3

DED1 BCD-Zahl in 1-Byte-Dualzahl
DED2 BCD-Zahl in 2-Byte-Dualzahl.

Ergänzungsmodule

SEMO selbständige Ergänzungsmodule mit Parameterübergabe.

Unter der Bezeichnung SEMO können weitere Module eingeführt werden, ohne in einen abgeschlossenen Compiler für PROLOG 2 eingreifen zu müssen. Die eigentliche Modulprozedur wird in die Sammlung von Sprachmoduln des Betriebssystems der SPS aufgenommen.

Betriebssystemrufe

STRT Start eines Programms zum einmaligen Lauf
STRZ Start eines Programms zum zeitzyklischen Lauf
CANC Austragung aus Zeit- und/oder Vorrangorganisation
PAUS Programmunterbrechung
GOTO Programmfortsetzung
DISP Programmverhinderung
ENAP Programmerlaubnis.

Programmierbeispiel

Es sei angenommen, daß der Steueralgorithmus, für den die SPS programmiert werden soll, bereits als Logikplan vorliegt. Im Bild A2.6 ist eine Teilschaltung (Auszug vom Gesamtlogikplan) dargestellt, für die hier das Steuerprogramm aufgestellt wird. Unter Berücksichtigung der byteweisen Zusammenfassung von Eingangs- und Ausgangssignalen durch die Prozeß-Signalein-/-ausgabe-Baugruppen werden die Eingänge E und Ausgänge A bezeichnet. Alle Zeit- und Zählbausteine werden klassifiziert und äquivalente

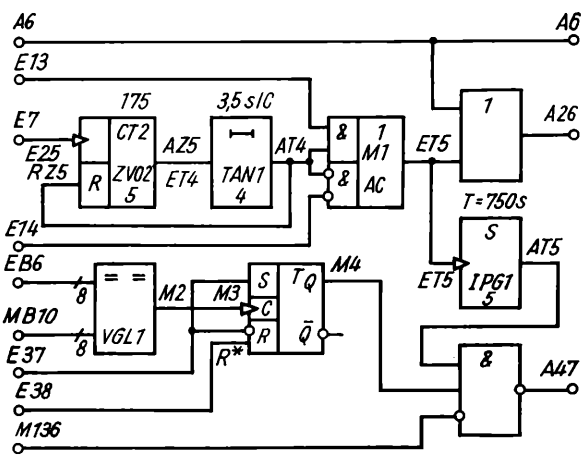


Bild A2.6
Logikplan einer Teilschaltung

Zeit- und Zählmodule von PROLOG 2 oder notwendigenfalls Ersatzschaltungen zugeordnet. Alle Zeitmodule werden von Ø beginnend durchnumeriert, ebenso alle Zählmodule. Alle Ein- und Ausgänge dieser Module werden mit den typspezifischen Variablenbezeichnungen versehen. Jedem durch Bausteine begrenzten Signalpfad einschließlich seiner Verzweigungen wird ein Bitmerker zugeordnet, sofern der Signalpfad nicht schon durch den Ein- oder Ausgang eines angrenzenden Zeit- oder Zählbausteins bezeichnet ist. Ein solcher Bitmerker kann, nachdem seine Weiterverarbeitung in den nachfolgenden Bausteinen erfolgt ist, an anderer Stelle erneut verwendet, d.h. also mehrfach verwendet werden. Das Ergebnissignal einer logischen Verknüpfung, das nicht über einen Knoten

verteilt wird, kann auch mit AC bezeichnet werden, sofern die im Logikplan nachfolgende Verknüpfung auch im Quellprogramm in der nächsten Zeile sofort erfolgt. AC repräsentiert den Akkumulator der Einzelbitverarbeitung und kann in der Programmnotation als temporärer Bitmerker verwendet werden.

Die zusätzlichen, für die Programmnotation notwendigen Bezeichnungen sind im Bild kursiv eingetragen. Der Rücksetzeingang des anzugverzögernden Zeitglieds TAN1 Nr. 4 ist nicht beschaltet; er wird auch in der Programmnotation nicht verwendet. Die Anzugverzögerungszeit von 3,5 s wird als 35 Dezisekunden angewiesen. Da der Impulsgeber IPG1 in jedem Zeitbereich nur bis maximal 255 programmiert werden kann, werden die geforderten 75Ø s als 125 Deziminuten angewiesen. Sollte die damit erreichte Auflösung nicht ausreichen, müßte nach einer Ersatzschaltung programmiert werden.

Das im Logikplan enthaltene positiv-flankengetriggerte RS-Flipflop muß durch eine Ersatzschaltung programmiert werden. Für PROLOG 2 wurde auf die Übernahme von Elementarspeichern in das Modulsortiment verzichtet, da sich dafür die Notation der äquivalenten Schaltgleichungen anbietet. Für synchron getaktete Elementarspeicher kann die Taktung bei einer SPS-Implementierung der Steuerschaltung meist durch den zyklischen Programmlauf ersetzt werden, bzw. sie erübrigt sich. Bei asynchron getakteten Elementarspeichern muß das Taktsignal in den Ansteuerbedingungen berücksichtigt werden. Im vorliegenden Fall muß vom Ergebnissignal M2 des Bytevergleichs der Ø-1-Übergang (positive Signalflanke) erkannt werden.

Die Anweisungsfolge lautet:

```
UMBI M2,M3 ;          M3 wird als Zwischenmerker verwendet
VGL1 EB6,MB1Ø,M2 ;    M2 wird neu erzeugt
UNDB /M3,M2,M3 ;      M3=1 bei positiver Signalflanke.
```

M3 wird nachfolgend mit den Ansteuerbedingungen für den Elementarspeicher konjunktiv verknüpft. Die Speichergleichung für das RS-Flipflop mit dominierendem Setzen (S-Flipflop) lautet:

$$^{k+1}Q = {}^{k}(S + \overline{R^*}Q).$$

Mit $S = E37 \cdot M3$ und $R^* = E38 \cdot M3$ folgt für die Notation der Speichergleichung mittels Sprachmodul BOOL:

BOOL $M4 = E37 \cdot M3 + /(E38 \cdot M3) \cdot M4$

Das Programm für die Schaltung, in der die vorgegebene Teilschaltung enthalten ist, soll zeitzyklisch mit 3Ø ms Zykluszeit abgearbeitet werden und im Rahmen aller SPS-Programme mit der Priorität 12 laufen. In der nachfolgenden Programmnotation gibt es eine Besonderheit, die nicht im Logikplan enthalten ist. Wenn erstmalig A6 = 1, wird der Grenzwert für Zählmodul Nr. 5 auf den Wert 35Ø gesetzt.

Programm

```
ØØØ1   APRO 12,3,CS ;        Programm der Priorität 12,
                             zyklisch mit 3Ø ms
ØØØ2   PEAV                  Prozeß-Ein-/Ausgabe-Vereinbarung
        ⋮
     k PRIN ... ;            Aufruf der Eingaben
        ⋮
     l UMBI E7,EZ5
```

```
l+ 1 UMBI  AT4,RZ5
l+ 2 UMD2 35Ø,GZ5,A6 ;                    GZ5:=35Ø, wenn A6=1
l+ 3 ZVO2  5,175 ;                        Vorwärtszähler Nr. 5
l+ 4 UMBI  AZ5,ET4
l+ 5 TAN1  4,35,DS ;                      Zeitmodul Nr.4
l+ 6 UNDB E13,AT4,M1
l+ 7 UNDB /AT4,/E14,AC ;                  Ergebnis bleibt im Akku, weil
l+ 8 ODRB  AC,M1,ET5 ;                    erster Operand in nächster Zeile
l+ 9 ODRB  A6,ET5,A26
l+1Ø IPG1  5,125,DM ;                     Zeitmodul Nr.5
l+11 UMBI  M2,M3 ;                        M3 puffert altes Vergleichsergebnis
l+12 VGL1  EB6,MB10,M2 ;                  Byte-Vergleich
l+13 UNDB /M3,M2,M3 ;                     M3=1 bei positiver Signalflanke
l+14 BOOL M4=E37.M3+/(E38.M3).M4 ;        S-Flipflop
l+15 UNDB AT5,M4,/M136,/A47
       ⋮
   m PROU ... ;                           Aufrufe der Ausgaben
       ⋮
   n PEND ;                               Ende des prioritätsgesteuerten
                                          Programms
```

Das gesamte Programm besteht aus n Vereinbarungen und Anweisungen. Die Teilschaltung wird durch 16 Anweisungen bzw. Modulaufrufe realisiert. Die logischen Grundverknüpfungen in den Programmzeilen $l + 6$ bis $l + 8$ können zusammengefaßt als Schaltgleichung programmiert werden. Mit

$$\text{BOOL ET5} = \text{E13} \cdot \text{AT4} + /\text{AT4} \cdot /\text{E14}$$

reduziert sich die Anweisungsfolge auf 14 Modulaufrufe.

Literaturverzeichnis

[2.1] *Stute, G.:* Steuerungstechnik. Berlin/Heidelberg/New York: Springer-Verlag 1981

[2.2] *Becker, H.; Leusch, G.:* Freiprogrammierbare Steuerung für Industrieroboter. Siemens-Zeitschrift 51 (1977) H.6, S.463–467

[2.3] *Haferkorn, P.; Schwarz, W.:* Steuerungen für Industrieroboter. Feingerätetechnik 30 (1981) H.7, S.9–13

[2.4] *Natschew, G.:* Steuerungsstrukturen adaptiver Roboter. Vortrag auf der Internationalen Konferenz (RGW) „Probleme der Steuerung von Industrierobotern", Varna/VR Bulgarien, 26. bis 31.Oktober 1981

[2.5] *Spur, G.; Auer, B. H.; Sinning, H.:* Industrieroboter – Steuerung, Programmierung und Daten von flexiblen Handhabungseinrichtungen. München/Wien: Carl Hanser Verlag 1979

[2.6] Autorenkollektiv: Industrieroboter. Herausgegeben von Prof. Dr.-Ing. habil. *J. Volmer.* Berlin: VEB Verlag Technik 1981

[2.7] *Sandner, W.:* Steuerungen für Industrieroboter – Möglichkeiten zur Systematisierung nach Steuerungsart und Programmierverfahren. Maschinenbautechnik 29 (1980) H.7, S.301–305

[2.8] *Blume, Ch.; Dillmann, R.:* Struktur und Programmierung von Industrierobotern, Teil 1 und 2. VDI-Zeitschrift 122 (1980) Nr.5, S.159–165, und Nr.6, S.231–239

[2.9] *Haferkorn, P.; Schwarz, W.:* Steuerungskonzepte für Industrieroboter. msr 24 (1981) H.5, S.286 bis 288

[2.10] *Swaczina, K.:* Industrieroboter. Elektronik (1981) H.24, S.78–81

[2.11] *Schraft, R. D.; Schiele, G.:* Industrieroboter zum Lackieren. I-Lack/50 (1982) H.2, S.65–73

[2.12] *Ignat'ev M. B.; Kulakov, F. M.; Pokrovskij, A. M.:* Algoritung upravlenijen robotani – manipulkatorani. Leningrad: Masinostroenie 1977

[2.13] *Weck, M.; D'Souza, C.:* Modulare Mehrprozessorsteuerung für Handhabungsgeräte in Fertigungsanlagen. Industrieanzeiger 102. Jg. Nr.1 vom 4.1.1980, S.16–20

[3.1] Autorenkollektiv: Industrieroboter. Berlin: VEB Verlag Technik 1981

[3.2] *Töpfer, H.; Kriesel, W.:* Funktionseinheiten der Automatisierungstechnik. 4. Aufl. Berlin: VEB Verlag Technik 1983

[3.3] *Schönfeld, R.; Habiger, E.:* Automatisierte Elektroantriebe. 2.Aufl. Berlin: VEB Verlag Technik 1983

[3.4] *Blackburn, J. F.; Reethof, S.; Shearer, S. L.:* Fluid Power Control. Wiesbaden: Krausskopf-Verlag 1962

[3.5] *Schulze, M.; Emmerling, F.; Haberland, K.:* Elektrische Achsantriebe für Industrieroboter. Wissenschaftlich-technische Informationen des VEB Kombinat Automatisierungsanlagenbau 18 (1982) 5, S.210–213

[3.6] *Stollberg, H.:* Linearachsen pneumatischer Industrieroboter. Maschinenbautechnik 29 (1980) 9, S.392–395

[3.7] *Schmalenbach, E.:* Elektropneumatische Antriebe für Industrieroboter. Werkstattechnik 69 (1979) 4, S.207–2 0

[3.8] *Littmann, K.:* Leistungsfähiger mikrorechnergesteuerter Servoantrieb. 4. Fachtagung Hydraulik und Pneumatik Magdeburg 1981

[3.9] *Sahner, G.:* Digitale Meßverfahren. 2. Aufl. Berlin: VEB Verlag Technik 1981

[3.10] *Bretschi, J.:* Intelligente Meßsysteme zur Automatisierung technischer Prozesse. München/Wien: R. Oldenbourg Verlag 1979

[3.11] *Tränkler, H.-R.:* Die Technik des digitalen Messens. München/Wien: R. Oldenbourg Verlag 1976

[3.12] Firmenschrift: Längemeßsysteme, Inkrementale Drehgeber und Code-Drehgeber. Heidenhain GmbH

[3.13] Induktives Wegmeßsystem. Werkstatt und Betrieb 110 (1977) 11, S.756

[3.14] Firmenschrift: Philips Linear-Meßsystem LMS IV

[3.15] *Paul, R.:* Mikroelektronik. Berlin: VEB Verlag Technik 1981
[3.16] Firmenschrift WF-Informationen: CCD-Zeile L110C Berlin 1980
[3.17] *Köhler, G. W.:* Industrie-Roboter mit Werkstück-Erkennungssystem – ein Überblick. technica 25 (1979) S.2183
[3.18] *Goldgewicht, W.:* Un dispositif d'identification de formes. La Machine Moderne 68 (1974) 6, S.22–25
[3.19] *Kuist, Ch. H.:* Those flexible, formable current-carrying plastics. Automation, Okt. 1976, S.54–57
[3.21] *Schwarz, A.; Schrepel, D.:* Pneumatische Sensoren für Handhabeeinrichtungen. Tagungsmaterial der 3. Wiss. Konferenz der IH Zwickau, März 1979
[3.22] *Gudat, H.:* Möglichkeiten akustischer Güteprüfung in der industriellen Fertigung. Fachberichte Messen, Steuern, Regeln, Band 1, Interkama-Kongreß 1977

[4.1] Autorenkollektiv: Robotertechnik, Bd. 1. Suhl: Bezirksverband der Kammer der Technik 1981
[4.2] *Blume, C.; Dillmann, R.:* Frei programmierbare Manipulatoren, Aufbau und Programmierung von Industrierobotern. Würzburg: Vogel-Verlag 1981
[4.3] *Vukobratović, M.; Potkonjak, V.:* Dynamics of Manipulation Robots. Berlin/Heidelberg/New York: Springer-Verlag 1982
[4.4] *Vukobratović, M.; Stokić, D.:* Control of Manipulation Robots. Berlin/Heidelberg/New York: Springer-Verlag 1982
[4.5] *Paul, R. P.:* Modeling, Trajectory Calculation and Servoing of a Computer Controlled Arm. Stanford Artificial Intelligence Laboratory, AIM 177, 1972
[4.6] *Horn, B.; Raibert, M.:* Configuration Space Control. Cambridge, Mass.: Artifical Intelligence Laboratory, Technical Report 1978
[4.7] *Pawlow, B.A.; Timofejew, A. B.:* Berechnung und Stabilisierung der programmierten Bewegung eines beweglichen Roboters (russ.). Technitscheskaja Kibernetika (1976) 6, S.91–101
[4.8] *Sinning, H.:* Steuerung und Regelung der Bewegungsachsen von Handhabungseinrichtungen. München: Carl Hanser Verlag 1980
[4.9] *Freund, E.; Hoyer, H.:* Das Prinzip nichtlinearer Systementkopplung mit der Anwendung auf Industrieroboter. Regelungstechnik 28 (1980) 3, S. 80–87, 4, S. 116–126
[4.10] *Budó, A.:* Theoretische Mechanik. 9. Aufl. Berlin: VEB Deutscher Verlag der Wissenschaften 1978
[4.11] *Schwarz, H.:* Hierarchische Regelkonzepte am Beispiel eines mehrrechnergeregelten industriellen Handhabungssystem. Ilmenau: 27. Internationales Wissenschaftliches Kolloquium 1982, A1/2, S.111–114
[4.12] *Sage, A. P.:* Optimum systems control. Englewood Cliffs, N.J.: Prentice-Hall 1968
[4.13] *Athans, M.; Falb, P. L.:* Optimal Control. New York/St. Louis/San Francisco: McGraw Hill 1966
[4.14] *Athani, V. V.:* Time Optimal Controller for Thyristor Controlled DC Motor Drive. Preprints: Control Science and Technology for the Progress of Society, Kyoto, Japan: IFAC 1981
[4.15] *Oppelt, W.:* Kleines Handbuch technischer Regelvorgänge. Berlin: VEB Verlag Technik 1964
[4.16] *Ackermann, J.:* Abtastregelung. Berlin: Springer-Verlag 1972
[4.17] *Zypkin, Ja. S.:* Grundlagen der Theorie lernender Systeme. Berlin: VEB Verlag Technik 1971
[4.18] *Bellmann, R.:* Adaptive Processes – A Guides Tour. Princeton: University Press 1961
[4.19] *Astroem, J.:* Introduction to Stochastic Control Theory. New York: Academic Press 1970
[4.20] *Feldbaum, A.A.:* Optimal Control Systems. New York: Academic Press 1965
[4.21] *Landau, Y. D.:* Adaptive Control – the Model Reference Approach. New York: Marcel Dekker 1979
[4.22] *Laerdal, S.:* Trallfa Robot – A new Concept of the Control System. 6th International Symposium in Industrial Robots, Nottingham, U. K. 1976
[4.23] *Oetken, B.:* Ein Beitrag zur Interpolation mit digitalen Filtern. Dissertation an der Universität Erlangen-Nürnberg 1978
[4.24] *Haugan, K. M.:* Continuous Path Controlled Robot in Arc Welding Applications. 6th International Symposium on Industrial Robots, Nottingham, U. K., 1976
[4.25] *Richiardi, C.:* Computerized Control of a Continuous Robot. 3rd CIRT/6th ISIR. University of Nottingham, U. K., 1976
[4.26] *Schraft, R. D.; Schleie, G.:* Industrieroboter zum Lackieren. I-Lack 50 2/1982, S.65–73
[4.27] *Llibre, M.*, u.a.: Data Compression Methods for the Recording of Industrial Robots Trajectories 12 th International Symposium on Industrial Robots. Paris 1982
[4.28] *Schwarz, W.; Geiler, J.; Fiß, P.:* Bewegungssteuerung bei Industrierobotern mit direkter Teach-in-Programmierung. msr (in Vorbereitung)
[4.29] *Wiener, N.:* Extrapolation, interpolation and smoothing of stationary time series. New York: John Wiley & Sons 1950

[4.30] *Denavit, J.:* A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. ASME Journal of Applied Mechanics 1953, p.215
[4.31] *Stackhouse, T.:* A new Concept in Robot Wrist Flexibility. Cincinnate Milacron Inc. (USA 1979)
[4.32] *Stoitschew, L.:* Eine Mikroprozessorimplementierung schneller Koordinatentransformation für Industrieroboter in Gelenkaufbau. Zmsr 24, 1981, H.5, S.289
[4.33] *Luh, J.Y.:* Evaluation on the Motion of Mechanical. Arm. Modeling and Simulation Vol. 7, 1976, p.220
[4.34] *Kiesewetter, H.:* Elementare Methoden der numerischen Mathematik. Berlin: Akademie-Verlag 1974
[4.35] *Spielberg, K.:* Efficient Continued Fraction Approximations to Elementary Functions. Math. Comp. Vol. 15, 1961, p.409
[4.36] *Baker, P.:* Suggestion for a Fast Binary Sine/Cosine Generator, IEEE Trans. on Comp. Vol. C 25, 1976, p.1134
[4.37] *Volder, J.:* The CORDIC Trigonometric Computing Technique. IRE Trans. on Electr. Comp., Vol. EC-8, 1959, p.330
[4.38] *Steer, O.:* Digital Hardware for Sine/Cosine Functions IEEE Trans. on Comp., Vol. C 26, 1977, p.1283
[4.39] *Haviland, G.:* A CORDIC-Arithmetic-Processor-Chip. IEEE Journ. on Solid State Circuit. Vol. SC-15, 1980, p.4
[4.40] *Paul, R.; Park, W.:* Manipulator Path Control by Microcomputer. Stanford Research Center, Industrial Robots Report 6, Nov. 1976
[4.41] *Whithey, D. E.:* The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators. Trans. ASME, 1972, Nr.12, p.303
[4.42] *Shimano, B. E.:* The Kinematic Design and Force Control of Computer Controlled Manipulators. Stanford University, AIM 313, 1978
[4.43] *Paul, R.:* Robot Manipulators: Mathematics, Programming, and Control. Cambridge: The MIT Press 1981

[5.1] *Gottschalk, W.:* SSP. Eine Methode zum strukturierten Systementwurf von Steuerungen für diskrete technische Prozesse.
[5.2] *König, H.:* Entwurf und Strukturtheorie von Steuerungen für Fertigungseinrichtungen ISW 13. Berlin/Heidelberg/New York: Springer-Verlag 1976
[5.3] *Schubert, W.:* Entwurfsrichtlinie: Systematischer Entwurf industrieller Steuerungen. Forschungszentrum Werkzeugmaschinen im Kombinat „Fritz Heckert", Karl-Marx-Stadt 1982
[5.4] *Krapp, M.:* Sequentielle Schaltung mit mehr als zwei Zuständen und beliebiger Topologie. WP 114 734, Kl. 21a1/1, 36/18, 22. 7. 1974
[5.5] *König, R.; Döhring, G.; Hummitz, P.; Quäck, L.:* Standardisierbarer Speicherbaustein mit *n* Zuständen und vollständigem Zustandsgraphen. WP 130 298, Kl.2(51), H.0.3K 19/00, 14.3.1977
[5.6] *Schubert, W.; Stoll, M.:* Verfahren und Schaltungsanordnung zur Vermeidung dynamischer Fehlschaltungen bei Zustandswechsel in digitalen Prozeßsteuerungen. WP 147011, Kl. G 05 B 19/00, 31.10.1979
[5.7] ursalog 4000, Baugruppenkatalog, Ergänzungssortiment. IfR Berlin, 1979
[5.8] *Meyer, G.; Fensch, S.:* Beschreibung, Analyse und Implementierung von software-realisierten Steuerungen durch Petri-Netze. msr 24 (1981), H.9
[5.9] *Starke, P. H.:* Petri-Netze. Berlin: Deutscher Verlag der Wissenschaften, 1980
[5.10] *Fensch, S.:* Ein Beitrag zur Beschreibung und Implementierung paralleler Abläufe in Steuerungssystemen. Dissertation, TH Karl-Marx-Stadt 1982
[5.11] *Schwarz, W.; Meyer, G.; Eckardt, D.:* Mikrorechner. Wirkungsweise, Programmierung, Applikation. Berlin: VEB Verlag Technik 1980
[5.12] *Quäck, L.; König, R.:* Zur Realisierung zeitmarkierter Petri-Netze auf der Basis der Corex-Struktur. ZKI-Informationen 1/19, Berlin 1978
[5.13] *Krapp, M.:* Modulare Struktur von Folgeschaltungen WP 105 331, Kl.42 r 1, 19/02, 30.7.1973
[5.14] *Meyer, G.; Weber, B.; Schubert, W.; Hiersemann, R.:* Fortsetzungsreihe: Industrieroboterstreuerungen msr 26 (1983), H.1, 2, 3, 4, 5, 6, 7, 8
[5.15] *Deutsch, K.-H.; Herklotz, G.; Kracht, E.; Scheibner, H.:* Richtlinie für die Einsatzvorbereitung des IR/S II mit der Steuerung IRS 2000. Nachnutzungsmaterial des Forschungszentrum des Werkzeugmaschinenbaus Karl-Marx-Stadt 1981
[5.16] *Schubert, W.; Fritzsch, P.; Hiersemann, R.:* Dokumentation zum Zwei-Maschinen-Beschickungsprogramm der technologischen Einheit IR/S II mit IRS 2000 und zwei PF 2 NC. Nachnutzungsmaterial des Forschungszentrums des Werkzeugmaschinenbaus Karl-Marx-Stadt 1981

[5.17] *Despang, G.; Arendt, F.; Quäck, L.:* SGM-Steuergraph-Mikro, eine Fachsprache im System ursalog 5020. Programmierhandbuch, 1983
[5.18] *Oberst, E.; Arendt, F.; Despang, G.; Haufe, J.:* Rechnergestützte Programmierung von Mikrorechnern auf der Basis von Steuergraphen (STEUERGRAPH-MICRO), ZKI-Informationen 2/80, Berlin 1980
[5.19] *Schubert, W.; Weber, B.:* Systematische Erarbeitung der Funktionsgleichungen für Werkzeugmaschinensteuerungen und deren Umsetzung in speicherprogrammierbare Steuerungen. msr 24 (1981), H.4
[5.20] *Weber, B.; Krumbiegel, B.; Zimmermann, B.:* Zur Programmierung von speicherprogrammierbaren Steuerungen für Werkzeugmaschinen, dargestellt an der Steuerung PC 600. msr 25 (1983), H.7
[5.21] *Zander, H.-J.:* Logischer Entwurf binärer Systeme Berlin: VEB Verlag Technik 1982
[5.22] *Busch, H. J.; Engelien, M.; Stahn, H.:* Algorithmisches System: Entscheidungstabellentechnik. Berlin: Akademie-Verlag 1981
[5.23] *Reißig, W.:* Petrinetze. Berlin/Heidelberg/New York: Springer-Verlag 1982
[5.24] *Valette, R.; Diaz, M.:* Top-down formal specification and verification of parallel controll systems. Digital Processes 4 (1978) 3/4, 181–199
[5.25] *Frank, M.; Lorenz, P.:* Simulation diskreter Prozesse. Leipzig: Fachbuchverlag 1979
[5.26] *Valette, R.:* An analysis oriented description of parallel systems allowing timing consideration. IFAC-Symp., Diskrete Systeme Dresden 1977, Bd.2, S.102–111

[6.1] *Schwarz, W.; Meyer, G.; Eckhardt, D.:* Mikrorechner. Wirkungsweise, Programmierung, Applikation. Berlin: VEB Verlag Technik 1980
[6.2] *Petermann, I.; Lang, R.; Gersch, W.; Junge, S.; Matern, R.:* Tendenzen der modularen Gestaltung von Systemen der Prozeßautomatisierung. Internat. Kongreß Metallbearbeitung IKM 82. Leipzig 1982
[6.3] *Paul, R. P.:* Modelling, Trajectory Calculation and Servoing of a Computer Controlled Arm. Stanford Artificial Intelligence Lab., Stanford University, CA, AI, Memo 177, Sept. 1972, USA
[6.4] *Paul, R. P.:* WAVE – A Model Based Language for Manipulator Control. The Industrial Robot. March 1977
[6.5] *Mujtaba, S.; Goldmann, R.:* Al User's Manual Stanford 1979, USA
[6.6] *Gini, G.; Gini, M.; Gini, R.; Giuse, D.:* Introducing Software Systems in Industrial Robots. Proc. 9th ISIR, Washington 1979
[6.7] User's Guide to VAL, a Robot Programming and Control System. Unimation, Inc., Danbury, Conn., 1980, USA
[6.8] *Dillmann, R.:* Weiterentwicklung der Steuerungs- und Bahnplanungsebene für Montageroboter mit höheren Programmiersystemen. VDI-Z 124 (1982) Nr.8
[6.9] *Auer, B. H.*, u.a.: Industrieroboter und ihr praktischer Einsatz. Grafenau: Expert-Verlag 1979
[6.10] *Blume, C.; Dillmann, R.:* Frei programmierbare Manipulatoren. Aufbau und Programmierung von Industrierobotern. Würzburg: Vogel-Verlag 1981
[6.11] *Spur, G.* (Hrsg.): Industrieroboter. München: Carl Hanser Verlag 1979
[6.12] Programmieranleitung ZIM 60/ZIM 10 – VEB Zentraler Ingenieurbetrieb der Metallurgie Berlin
[6.13] ZIM-Dokumentation – VEB Zentraler Ingenieurbetrieb der Metallurgie Berlin
[6.14] Bedienungsanleitung IRS 600. VEB NUMERIK „Karl Marx", Karl-Marx-Stadt 1981
[6.15] *Spur, G.; Auer, B. H.; Sinning, H.:* CNC-Control for Industrial Robots with a Modular Programming Language. 7th ISIR, Tokyo 1977
[6.16] *Carlisle, B.:* The PUMA/VS-100 Robot vision System. 1st Conf. on Rob. Vision and Sensory Controls., Stratford 1981
[6.17] *Inoue, H.; Ogasawara, T.; Shiroshita, O.; Naito, O.:* Design and Implementation of High Level Robot Language. 11th ISIR Tokyo 1981
[6.18] *Blume, C.:* A Structured Way of Implementing the High level Programming Language AL on a Mini- and Microcomputer Konfiguration. 11th ISIR Tokyo, 1981
[6.19] *Latombe, I. C.; Mazer, E.:* LM – High-level Programming Language for Controlling Assembly Robots. 11th ISIR Tokyo, 1981
[6.20] *Takase, K.; Paul, R.:* A Structured Approach to Robot Programming and Teaching. IEEE Trans. SMC-11, Nr.4, April 1981
[6.21] *Taylor, R. H.:* The synthesis of Manipulator Control Programs from Task-level Specifications. Diss. Stanford, 1976, USA
[6.22] *Lozano-Perez, T.; Winston, P.:* LAMA: A Language for Automatic Mechanical Assembly. Proc. 5th IJCAI, Boston, 1977

[6.23] *Liebermann, L.; Wesley, M.:* AUTOPASS: An Automatic Programming System for Computer Controlled Mechanical Assembly, IBM, J. Res. Dev. 21, 4, 1977
[6.24] *Popplestone, R.; Ambler, A.; Bellos, J.:* RAPT – A Language for Describing Assemblies. The Industrial Robot, 3, 1978
[6.25] *Lozano-Perez, T.:* Automatic Planning of Manipulator Transfer Movements. IEEE Trans. SMC-11, Nr.10, 1981
[6.26] *Barbera, A.:* An Architecture for a Robot Hierarchical Control System. Washington, 1977, USA
[6.27] *Bonner, S.; Shin, Kang, G.:* A Comparative Study of Robot Languages. IEEE Trans. Comp. Nr.12, 1982, p.82–96

[7.1] *Pugh, A.:* Second Generation Robotics. Proc. 12th Int. Symp. on Industrial Robots, Paris 1982
[7.2] *Masuda, R.; Hasegawa, K.:* Total Sensory System for Robot Control and its Design Approach. Proc. 11th Int. Symp. on Industrial Robots, Tokyo 1981
[7.3] *Volmer, J.* (Hrsg.): Industrieroboter. Berlin: VEB Verlag Technik 1981
[7.4] *Volmer, J.; Jacobi, P.; Haehnel, K.:* Fügemechanismen mit nachgiebigen Gliedern für Montage-Industrieroboter. Maschinenbautechnik 29 (1980), H.3
[7.5] *Dillmann, R.; Faller, B.:* Kraft-Momenten-Sensorsystem für Industrieroboter. Elektronik 8 (1982), H.4, S.89
[7.6] Firmenschrift FS 6-120 A. 6-Axis Force Sensor with Onboard Mikroprocessor. Astek Engineering, Watertown, USA 1983
[7.7] *Zecha, M.; Fritzsch, K.; Schwarz, A.:* Second Generation Robots – Research and Some Applications in the GDR. Proc. 11. Int. Symp. on Industrial Robots. Paris 1982, p.311
[7.8] Mehrachsige Meßdose. BRD-Patent 2826131 (1979)
[7.9] Method and apparatus for six degree of freedom force sensing. US-Patent 4094192 (1978)
[7.10] *Raibert, M. H.; Tanner, J. E.:* A VLSI Tactile Array Sensor Proc. 12th Int. Symp. on Industrial Robots, Paris 1982
[7.11] Tactile Matrix Sensor. Firmenschrift, Astek Engineering, Watertown, USA, 1983
[7.12] Mit induktiven Initiatoren berührungs- und kontaktlos schalten. VEB Meßgerätewerk Beierfeld 1981
[7.13] *Schulze, K. R.; Faber, W.:* Sensoren für Lichtbogenschweißroboter und -automaten. ZIS Halle, 1982
[7.14] *Dünnebier, G.*, u.a.: Positionsempfindliche Fotoempfänger Feingerätetechnik 30 (1981), H.4, S.170
[7.15] *Bamba, T.*, u.a.: A Visual Sensor für Arc-Welding Robots. Proc. 11th Int. Symp. on Industrial Robots, Tokyo 1981
[7.16] *Purbrick, J.:* A Force Transducer Employing Conductive Silicone Rubber. Proc. 1. Conf. on Robot Vision and Sensory Control, Stratford 1981, p.73
[7.17] Firmenschrift L 110 C. VEB Werk für Fersehelektronik, Berlin 1981
[7.18] *Barbe, D. F.:* Charge-Coupled Devices. Berlin/Heidelberg/New York: Springer-Verlag 1980
[7.19] *Pratt, W. K.:* Median Filtering. Semiannual Report, Image Processing Institute, Univ. of Southern California, Sept. 1975, p.116
[7.20] *Fritzsch, K.:* Anwendung von Verfahren der Bildverarbeitung und Bilderkennung für Industrieroboter. Forschungsbericht des ZKI der AdW, 1979
[7.21] *Huang, T. S.:* A Fast Two-Dimensional Median Filtering Algorithm. IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. ASSP – 27, Nr.1, 1979, p.13
[7.22] *Ostsu, N.:* A Threshold Selection Method from Gray-Level Histogramms. IEEE Trans. Syst. Man and Cybern. vol. SMC-9, 1979, Nr.1, p.62
[7.23] *Fritzsch, K.; Roesler, U.:* Bilderkennung für Industrieroboter und visuelle Inspektion. Bild und Ton, 35 (1982) H.9, S.263
[7.24] *Lanz, O. E.:* Sensoren zur Lage- und Formerkennung. Interkama-Kongreß, Bd.1, 1977, S.95
[7.25] *Chow, C. K.:* On Optimum Recognition Error and Reject Tradeoff. IEEE Trans. I T-16, 1970, p.41
[7.26] *Simon, F.* (Hrsg.): Automatische Bildverarbeitung in Medizin und Biologie. Dresden 1975
[7.27] *Agin, G. J.:* An Experimental Vision System for Industrial Application. Proc. 5th Intern. Symp. on Industrial Robots, Chicago 1975, p.135
[7.28] *Koskinen, K.:* Object Recognition and Handling in an Industrial Robot System with Vision. Proc. 8th Intern. Symp. on Industrial Robots, Stuttgart 1978, p.744
[7.29] *Armbruster, K.*, u.a.: A very Fast Vision System for Recognizing Parts and their Location and Orientation. Proc. 9th Intern. Symp. on Industrial Robots, Washington 1979
[7.30] *Bretschi, J.*, u.a.: Reduction of Information in Optical Sensors of Industrial Robots. Proc. 2nd Intern. CISM-IFToMM Symp., Warschau 1976, p.285

[7.31] *Roesler, U.; Klette, R.; Sommer, G.:* Digitale Bildverarbeitung in der Automatisierungstechnik. msr 26 (1983) H.11
[7.32] *Bretschi, J.:* Intelligente Meßsysteme zur Automatisierung technischer Prozesse. München: Oldenbourg Verlag 1979
[7.33] *Schwarze, G.; Vlcek, J.:* Realtime Image Processing for Robotic and Inspection Systems with Specialized Processors. Proc. 2nd Int. Conf. on Artificial Intelligence and Information-Control Systems of Robots. Smolenice 1982, p.220
[7.34] *Gleason, G. J.; Agin, G. J.:* A Modular Vision System for Sensor-controlled Manipulation and Inspection. Proc. of 9th Int. Symp. on Industrial Robots, Washington, 1979
[7.35] *Padberg, C.:* OMS-Optoelektronische Meß- und Sensortechnik. BBC Brown Boveri, Firmendokumentation, 1981
[7.36] *Fu, K. S.:* Robot Vision for Machine Part Recognition. Proc. SPIE Conf. 442, San Diego 1983
[7.37] *Fimmel, B.*, u.a.: A Vision System for real-time Handling and Recognition of Industrial Parts. Int. Conf. Artificial Intelligence and Information of Robots. Smolenice (ČSSR), 1980
[7.38] *Rosen, C. A.; Nitzan, D.:* Use of Sensors in Programmable Automation. Computer, Vol. 10 (1977) Nr.12, p.325
[7.39] *Kempe, V.:* Sensortechnik für Industrieroboter der 2. und 3. Generation. msr 26 (1983) H.1 u. 2

[8.1] Betriebsdokumentation Mikrorechnersystem K1520. VEB Robotron-Elektronik Zella-Mehlis
[8.2] *Fickel, W.; Kehrer, E.; Wollenberg, G.:* Numerische Steuerungen auf der Basis von Mikrorechnern. msr 21 (1978) H. 9, S.485 u. 486
[8.3] Linieninterface Bus K1520, TGL 37271
[8.4] *Wollenberg, G.:* CNC 600 – ein numerisches Steuerungssystem auf Mikrorechnerbasis. rfe 28 (1979) H.11, S.694–700
[8.5] *Kehrer, E.; Schröter, K.; Wollenberg, G.:* Systembeschreibung CNC 600. Internes Schulungsmaterial, VEB Numerik Karl-Marx-Stadt, 1981
[8.6] *Wollenberg, G.; Tißold, J.:* Mikrorechnersteuerungen für Be- und Verarbeitungsmaschinen – Stand und Entwicklungstrend. Wiss.-techn. Informationen des VEB Kombinat Automatisierungsanlagenbau 18 (1982) H.2, S.43–48
[8.7] Bahnsteuerungen System CNC 600, TGL 36173
[8.8] Technische Kundendokumentation CNC 600-1, Variante Bohren/Fräsen 451662-3 TK 4. VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.9] Technische Kundendokumentation CNC 600-1, Variante Drehen 451661-5 TK 4. VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.10] Steuerungen System CNC-H 600, TGL 36946
[8.11] CNC-H 600, Kundenprospekt, 1981
[8.12] Technische Kundendokumentation CNC-H 646, 451814-3 TK 4. VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.13] *Haufert, R.; Keßler, H.:* Das numerische Steuerungssystem CNC-H 600. Wiss.-techn. Informationen des VEB Kombinat Automatisierungsanlagenbau 17 (1981) H.1, S.38–42
[8.14] Programmierbare Steuerung PC 600, TGL 38759
[8.15] Technische Kundendokumentation Programmierbare Steuerungen PC 601, PC 602, PC 603, 444737-3 TK 4. VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.16] *Sobottka, D.; Stoll, M.; Schröter, K.:* PC 600 – eine neue Generation speicherprogrammierbarer Steuerungen. Wiss.-techn. Informationen des VEB Kombinat Automatisierungsanlagenbau 18 (1982) H.2, S.38–43
[8.17] Robotersteuerungen IRS 600, TGL 38758
[8.18] Bedienanweisung IRS 600, 444522-0 Ba 4. VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.19] *Barth, W.:* Industrierobotersteuerung IRS 600. Wiss.-techn. Informationen des VEB Kombinat Automatisierungsanlagenbau 17 (1981) H.1, S.43–45
[8.20] *Oehme, J.:* Programmierung, Bedienung und Einsatz der Robotersteuerung IRS 600. Wiss.-techn. Informationen des VEB Kombinat Automatisierungsanlagenbau 18 (1982) H.3, S.92–98
[8.21] *Bartsch, H.-J.; Wätzig, R.; Barth, W.:* Steuerungen der Industrieroboter. Technische Gemeinschaft 29 (1981) H.12, S.42–44
[8.22] CNC 603 – Freiprogrammierbare Steuerung des Systems 600. Kundenprospekt, VEB Numerik „Karl Marx", Karl-Marx-Stadt
[8.23] *Laerdal, S.:* Trallfa Robot – A new concept of control for Spray-Painting and Arc-Welding Applications. 6th International Symposium of Industrial Robots, Nottingham 1976
[8.24] *Shimon, A.:* Computer controlled Industrial Robot for Painting. 8th International Symposium of Industrial Robots, Stuttgart 1978

[8.25] Anleitung für den Programmierer/Systemprogrammierer – Geräteroutinen MOS K1520. VEB Robotron Zentrum für Forschung und Technik 1981

[8.26] *Haferkorn, P.; Schwarz, W.:* Steuerungen für Industrieroboter. Feingerätetechnik 30 (1981) 1, S.9–13

[8.27] *Schwarz, W.; Meyer, G.; Eckhardt, D.:* Mikrorechner. Wirkungsweise – Programmierung – Applikation. 2. Aufl. Berlin: VEB Verlag Technik 1982

[8.28] Sprachbeschreibung Assemblersprache SYPS K1520. Systemunterlagen – Dokumentation VEB Robotron Zentrum für Wissenschaft und Technik 1979

[8.29] Einheitsbaureihe Stromversorgungsbaugruppen, Anwendungsvorschrift. VEB Kombinat Robotron Zentrum für Forschung und Technik 1982

[8.30] ursatron 5000. Kundeninformation Kombinat EAW Berlin ursatron 5000; ursadat 5000 Bd.1 bis Bd. 5. Kundeninformation des VEB Kombinat EAW Berlin

[8.31] ursatron 5000; Betriebsdokumentation Programmiergerät ursatron 5000. Kundeninformation des VEB Kombinat EAW Berlin

[8.32] ursalog 4000. Prospektmaterial und Projektierungsrichtlinie des VEB Kombinat EAW Berlin

[8.33] ursalog 5010; Projektierungs-, Programmierungs- und Inbetriebnahmehinweise. VEB Kombinat EAW Berlin

[8.34] VEB Elektro-Apparate Werke Berlin-Treptow „Friedrich Ebert": Programmierhandbuch für SPS ursalog 5020

[8.35] *Kozariscuk, R.; Poppe, H.:* Fachsprache PN 5000 zur Beschreibung von steuerungs- und regelungstechnischen Aufgaben innerhalb der Anlagenautomatisierung. msr 24 (1981) H.5, S.273–276

Sachwörterverzeichnis

Ablaufkontrollnetze 165, 177ff.
Ablaufsteuerung 17, 21, 168ff.
Abtastfrequenz 116f.
Abtastung 116, 127
Achskoordinaten 77
Aktionssteuerung 18, 22, 149
AL 214
d'Alembertsches Prinzip 81
Antriebe 39ff.
–, elektrische 53ff.
–, elektrohydraulische 40ff., 113
–, pneumatische 62f.
–, Schritt- 63
Arbeitsspeicher 125f.
Arbeitszylinder, hydraulischer 113
Aufnehmer, inkrementaler 70
Ausgangsgleichung 150
Automat, endlicher 160
Automatengraph 156
Automatentabelle 153

Bahnsteueraufgabe 76
Bahnsteuerung 29, 123ff.
Bahnvektor 142
Bahnvermessung 223
Bang-Bang-Regelung 100f.
Basiskoordinaten 148
Bedingungs-Ereignis-Systeme 160
Beleuchtungstechnik 243
Betriebsart 25, 168ff.
Bewegungsgleichung 78, 86
–, Lagrangesche 81f.
Bewegungsplanung 148
Bewegungssteuerung 17, 21, 76ff.
Biasvektor 229
Bildvorverarbeitung 248
Binarisierung 246, 251

CAD 220
CCD-Kamera 244
CCD-Zeile 244
Corioliskraft 83, 86

Datendekompression 128
Datenkompression 127
Dehnmeßstreifenbrücke 230ff.
Deltamodulation 126ff.
Denavit-Hartenberg-Matrix 138f.
Dialogsteuerung 18, 25
Distanzmessung, optische 237

Entkopplungsmatrix 229
Entkopplungsstufe 96
Entscheidungstabelle 153
Entwurf
–, funktioneller 151, 167f.
–, intuitiver 152
–, struktureller 151, 184ff.
–, systematischer 152
–, technischer 151, 184ff.
Entwurfshierarchie 169f.
Entwurfsphase 150
Erkennungssysteme 241ff.
–, dreidimensionale 261
–, visuelle 241ff.
Euler-Zerlegung 81
Existenzkanten 163f.
Exklusivität 180

Fallgraph 160
Flächenträgheitsmoment 252f.
Fliehkraft 83
Folgeregelung 97f., 124
Frame 148
Freeman-Kette 257
Führungsgrößenerzeugung 123
Führungsgrößengenerator 117f.
Führungsverhalten 103
Funktionsinterpolatoren 133
Funktionspläne 155
Funktionssteuerung 21

Gelenkkoordinaten 142
Gesamtübertragungsfunktion 121
Geschwindigkeitsregler 99f., 114
Getriebe 57ff.
Gleichungen, logische 184f.
Grauwertverarbeitung 256
Gray-Kode 71
Greiferstellung 138
Greifkraftmessung 228
Grenzfrequenz 67
Grobentwurf 179
Grundgerät 13, 27

Harmonic Drive 57f.
Hauptachsen 82
Histogramme 251

Identifikation 120ff.
Implementierungsphase 151
Inductosyn 47f.
Inhibitorkanten 163f.
Intelligenz, künstliche 118

Interpolation 126ff.
–, lineare 128
–, quadratische 129, 147
–, zirkulare 147

Kalibrierung 260
Kette, kinematische 78
Kinematikglied 81
Klassifikator 253
–, paralleler 253
–, sequentieller 254
Konturbild 243
Konturverfolgung 225, 236
Koordinaten 133ff.
–, geräteeigene 133ff.
–, kartesische 133ff.
–, sphärische 136
–, zylindrische 136
Koordinatensystem, greiferfestes 137
Koordinatentransformation 133ff.
–, direkte 133ff.
–, inkrementelle 143
–, inverse 133ff.
Koordinierungssteuerung 178
Kreisinterpolation 147
Kreisverstärkung 116

Lageregelung 104ff.
–, adaptive 119ff.
–, digitale 112ff.
–, lineare 104ff.
Lagrangefunktion 82, 85
Lastvariation 122
Leckleitwert 93
Lichtschnittechnik 239
Logikplan 155

Manipulator 13
Manipulatorsprachen 206
Maschine, virtuelle 32
Maskenoperation 253
Master-Slave-Programmierung 201
Mehrkoordinaten-Meßsysteme 226
Merkmale, morpometrische 251, 257
Meßroboter 226
Meßsysteme 64ff.
Meßverfahren
–, absolute 70
–, inkrementale 71
–, phasenzyklische 73
Mikrorechner 29ff.
Modelle
–, globale 166
–, lokale 166f.
Modellbildung 166
Modelldaten 252
Modellierungsphase 150
Multipunktsteuerung 29, 124ff.
Mustervergleich 256ff.

Netze 158ff.
–, Prädikat-Transitions- 156
–, Stellen-Transitions- 161
Nullhubregelung 51

Ölkompressibilität 114f.
Ölstrom 113
Orientierung 138
Orientierungsmerkmal 253

PC 149
Petrinetze s. Netze
Phasenebene 100
Positionierdauer 100ff.
Positionierung, zeitoptimale 102f.
Positionierverhalten 79
Positionsgeber 70ff.
Präzisionspotentiometer 70
Programmablaufgraph 156
Programmiersprachen 214ff.
Programmiersystem 17, 216ff.
Programmierung 195ff.
–, externe 195
–, Folge- 200
–, interne 195
–, manuelle 196
–, Teach-in- 198
–, textuelle 204
Programmschleife 208
Punkt-zu-Punkt-Steuerung 28

Radialprofil 255
Rangordnungsoperator 248
Reaktionskräfte 112
Realisierungsphase 151f.
Referenzmodell 120
Reflexlichtschranke 234
Reglerstruktur 115
Reibung
–, Coulombsche 117
–, geschwindigkeitsproportionale 115
Relaisplan 155
Repräsentant 252f.
Resolver 73
Resonanzfrequenz, mechanische 121
Richtungskosinus 136
Roboter
–, Beschickungs- 14
–, technologischer 16
–, Montage- 17
Roboterprogramm 204f.
Rotationsmatrix 135

Schalenmodell 30
Schnittgrößen, mechanische 227ff.
Schrittantriebe 63
Schubglied 36
Schweißnahterkennung 240
Schwenkglied 37

Sensoren 65ff., 222ff.
–, äußere 80, 222
–, Distanz- 235
–, innere 80, 222
–, Näherungs- 233
–, taktile 230ff.
Servoregelung 95, 103ff.
Servoventil 44ff., 92, 113f.
Silhouettenbild 243
Sollwertfolge 125f., 143
Speichertiefe 126
Speicherwortbreite 126
Sprachebenen 32
Starrkörpermodell 88
Stellglieder, elektrische 56f.
Stellgröße 99
Stellmotoren 54f.
Steuernetze 163
Steuerungsrealisierung
–, speicherprogrammierte 152
–, verbindungsprogrammierte 152
Störkräfte 110

Tastsensoren 225
Teach-in-Phase 226
Teach-in-Programmierung 138, 198
–, direkte 198
–, indirekte 138, 201
Teilautomat 149
Transformation, homogene 134f.

Überschleifbewegung 167
Überschwingweite 106
Übertragungsfunktion 47
Übertragungsverhalten
–, von elektrischen Antrieben 59ff.
–, von hydraulischen Antrieben 48ff.
Umweltmodell 19, 22

VAL 212
Verformkörper 228
Verstellpumpe 51
Vorhersagefilter 131

Wandler 67
Wandlung, optisch-elektrisch 241
Wegmessung 68ff.
Wellgetriebe 57
Winkel, Eulersche 136
Winkelgeber 70

Zustandsebene 100ff.
Zustandsgleichung 150
Zustandstrajektorie 103